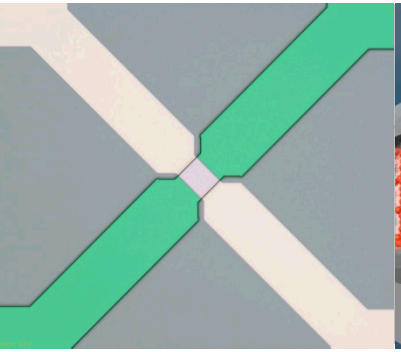
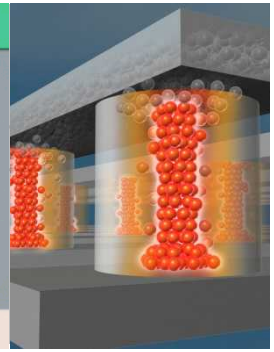
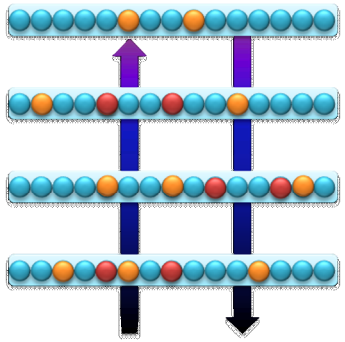
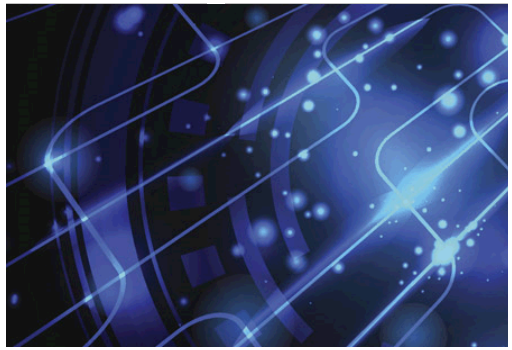
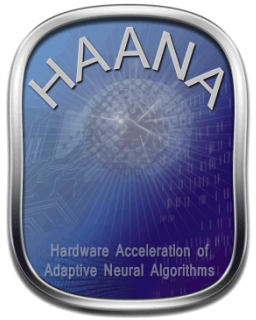
SAND2017-4198C

Neural-inspired computing algorithms and hardware for image analysis and cybersecurity applications

Conrad D. James, Ph.D.
Sandia National Laboratories

The Salishan Conference on High-Speed Computing, April 2017

Sandia National Laboratories is a multi-mission laboratory managed and operated by Sandia Corporation, a wholly owned subsidiary of Lockheed Martin Corporation, for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-AC04-94AL85000. SAND NO. SAND2017-XXXX

Acknowledgments

Algorithms: Brad Aimone, Ojas Parekh, Nadine Miner, Sandra Faust, Steve Verzi, Frances Chance, Tu-Thach Quach, Chris Lamb, Meghan Galiardi, Samuel Mulder, William Severa, Kristofor Carlson, Michael Smith, Cynthia Phillips, Jacob Hobbs, Robert Abbott; Jeffrey Piersol,

Architecture: John Naegle, Alex Hsia, Sapan Agarwal, Craig Vineyard, Fred Rothganger, Jonathon Donaldson, Gabriel Popoola, Aaron Hill

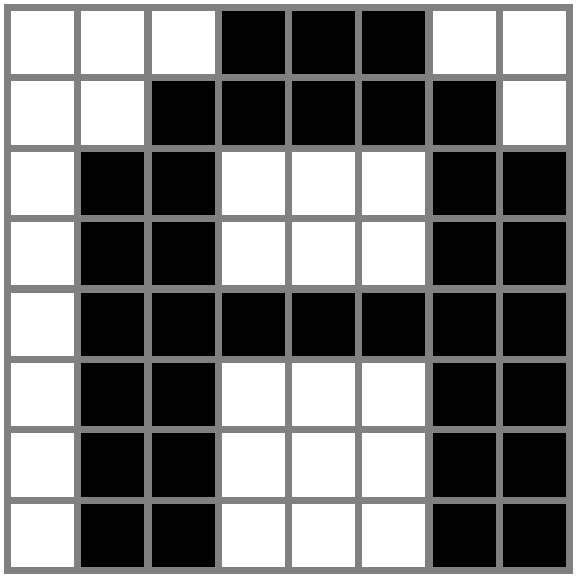
Learning Hardware: Matt Marinella, Thomas Beechem, Ron Goeke, Alec Talin, Paul Kotula, Farid El Gabaly, Elliot Fuller, Jim Stevens, David Hughart, Andy Armstrong, David Henry, Gaadi Haase, Steve Wolfley, Seth Decker, Christopher Saltonstall, Jamison Wagner, John Niroula, Derek Wilke, Michael Van Heukelom, Patrick Finnegan, Carl Smith, Robin Jacobs-Gedrim

Modeling & Applications: Steve Plimpton, **Justin Doak,** Richard Schiek, Brian Tierney, Robert Bondi, Harry Hjalmarson, Tim Draelos, Jonathan Cox, Joe Ingram, Jason Wheeler

Partnerships: David Follett, Duncan Townsend (Lewis Rhodes Labs); Pamela Follett; Isaac Richter, Engin Ipek (U. Rochester); Felix Wang (UIUC); David Lidsky, Marek Osinski (UNM);

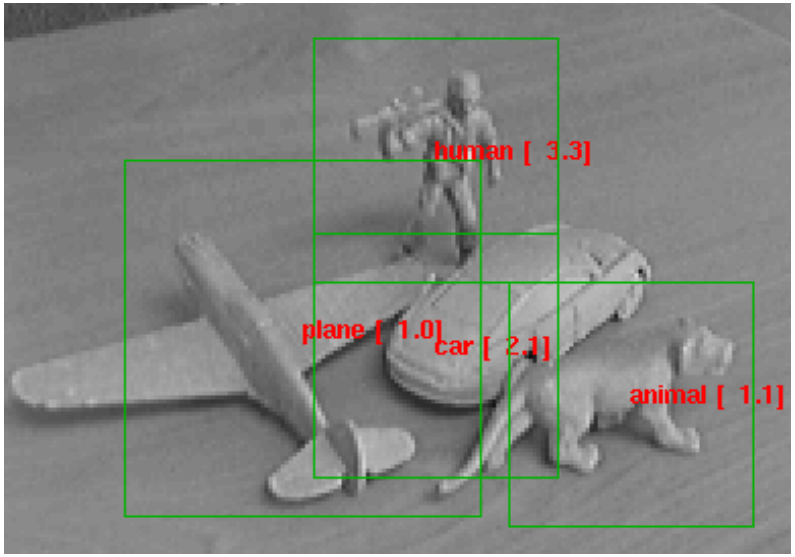
Data-driven computing (machine learning) is necessary for real-world problems

Conventional numerical computing



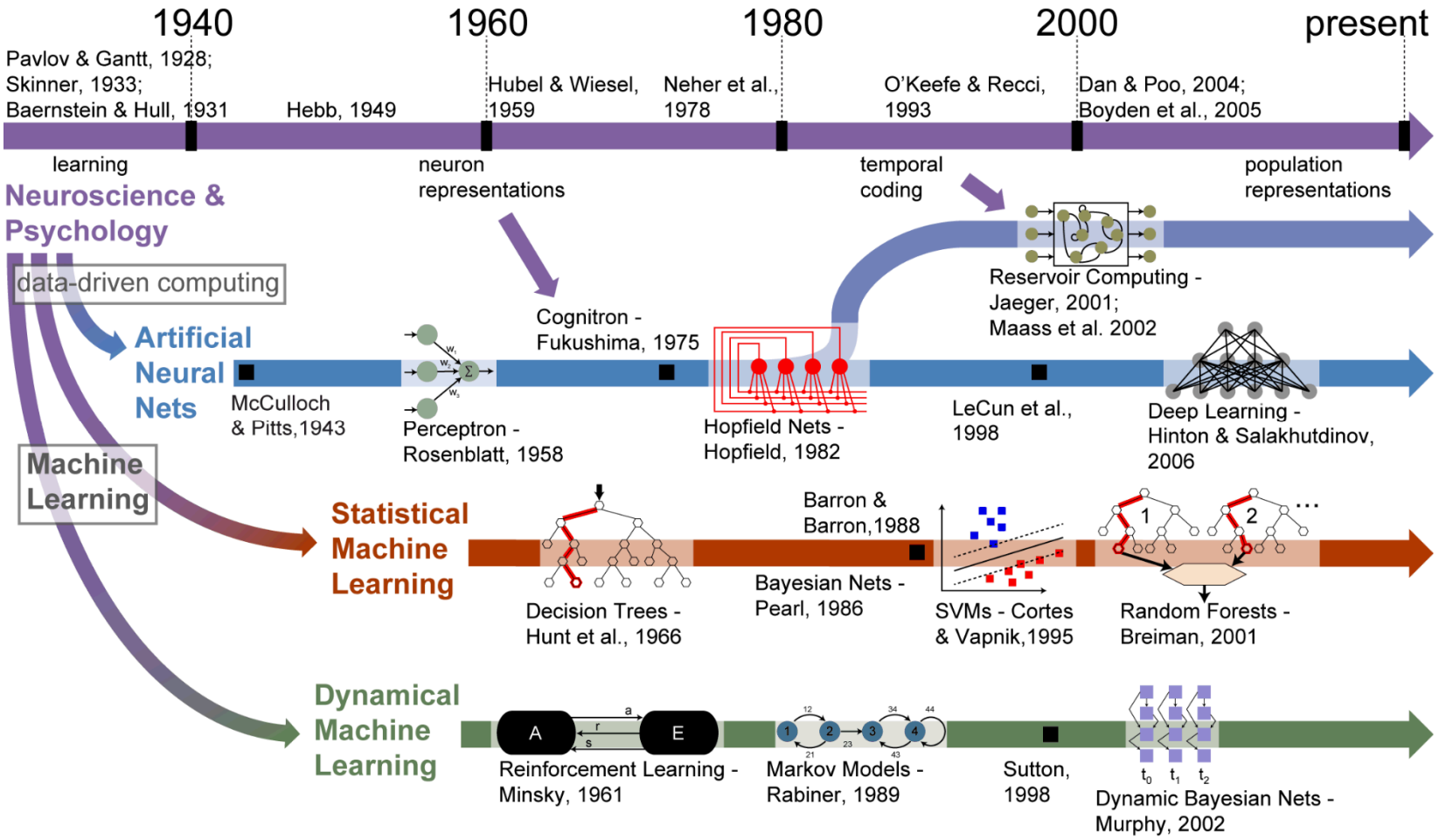
C. Lampert, VRML 2013

Data-driven computing



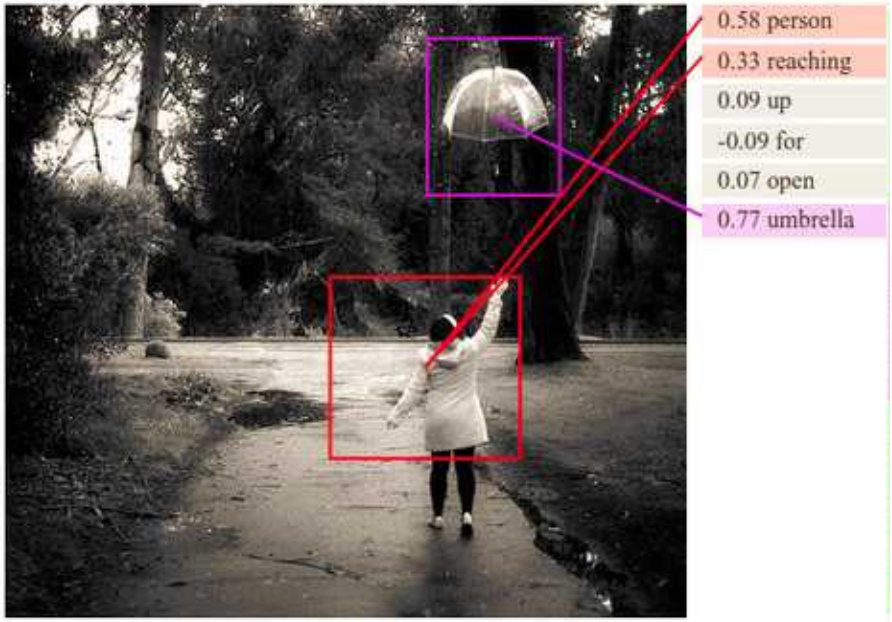
yann.lecun.com

Data-driven (neural-inspired) computing has a complicated history…and mixed results

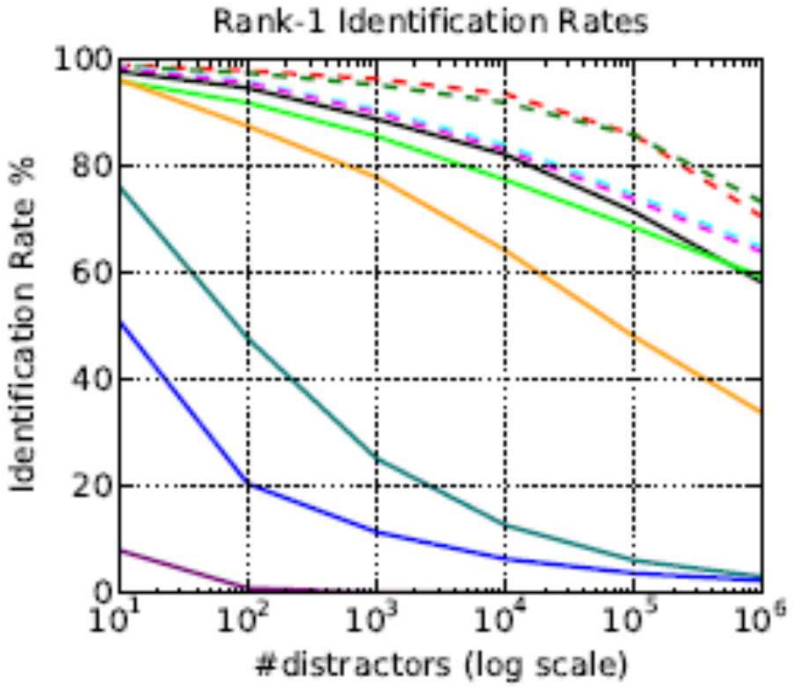


James et al., BICA 2017, 19, 49

Neural-inspired algorithms are achieving success but several challenges remain

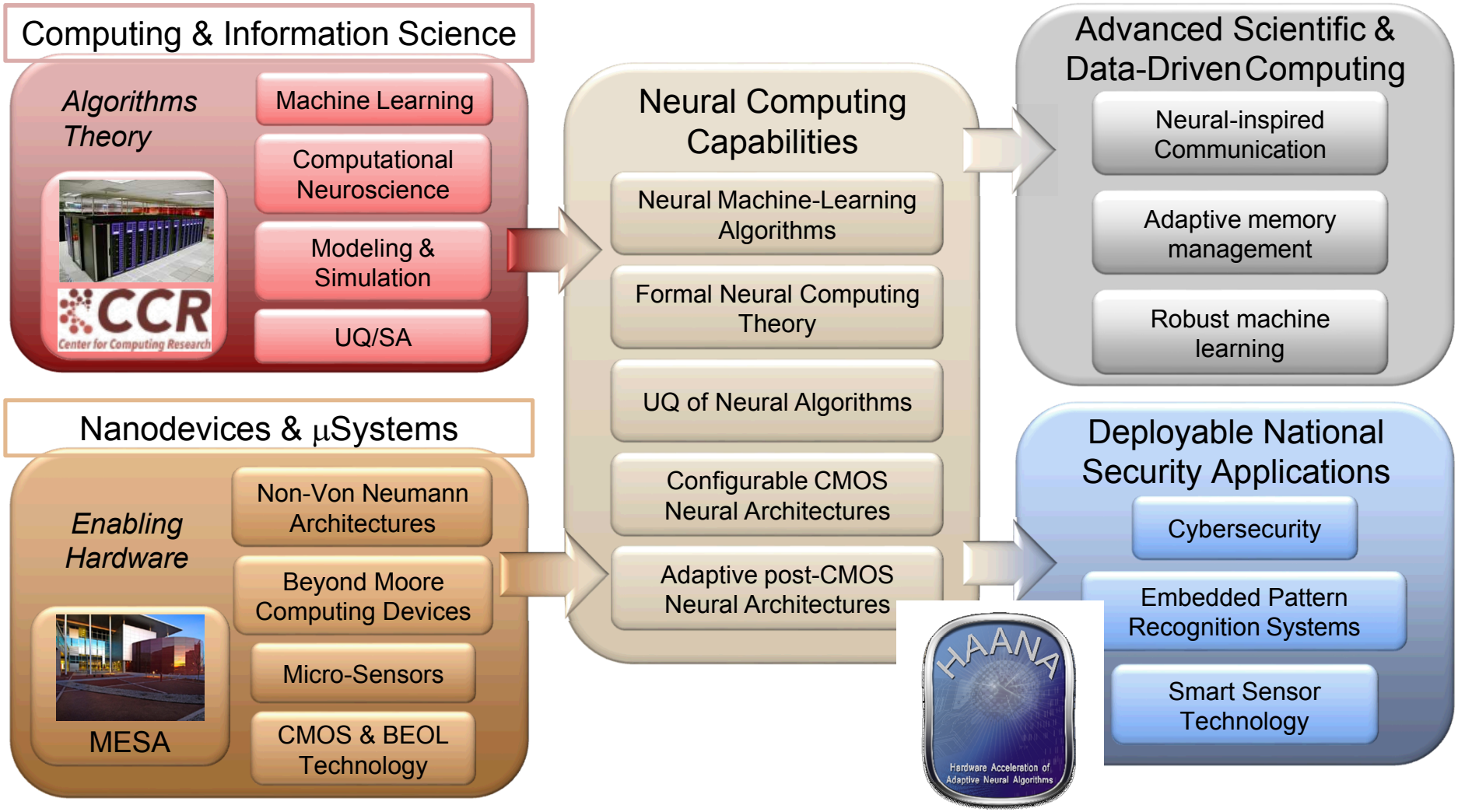


Karpathy etc. NIPS 2014, 1889

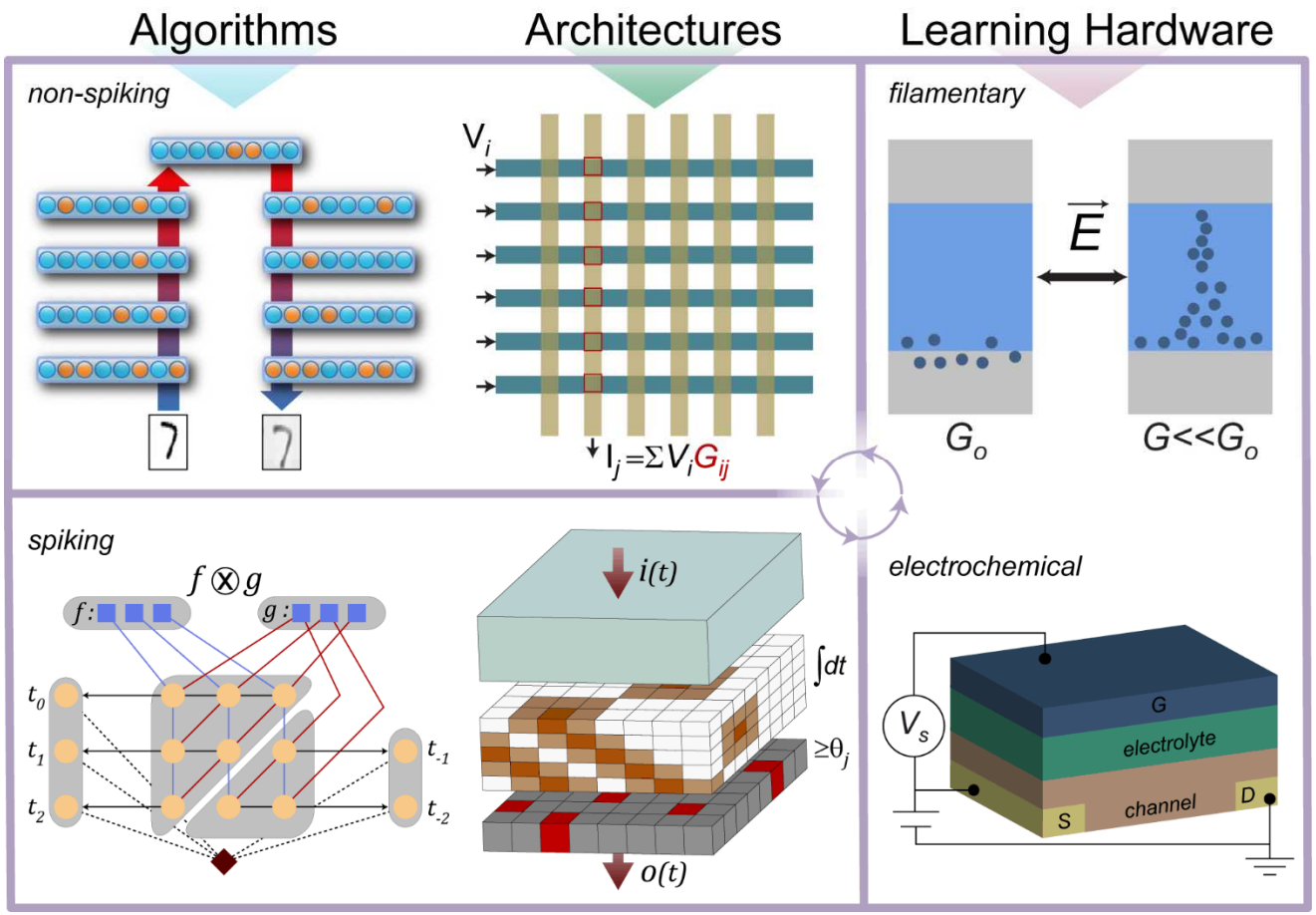


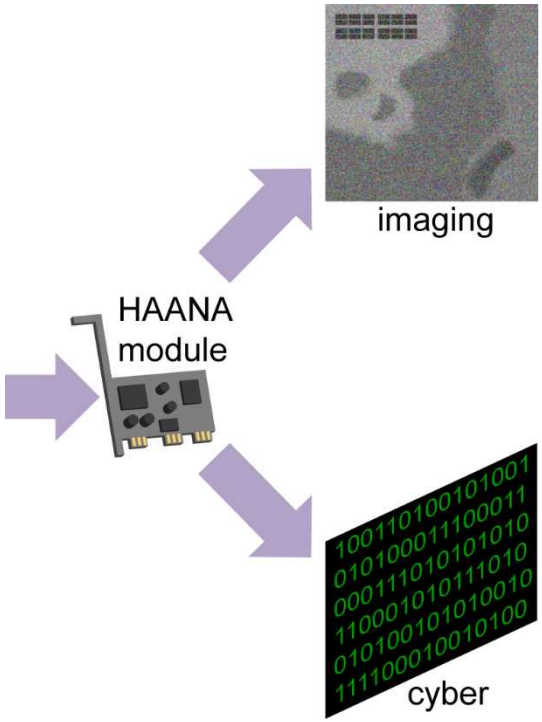
Kemelmacher et al., CVPR 2016

Neural computing at Sandia Labs leverages a large research foundation



Hardware Acceleration of Adaptive Neural Algorithms (HAANA)



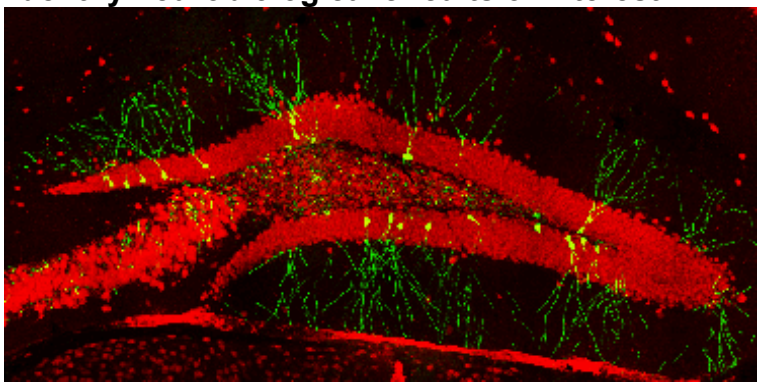


Neurogenesis deep learning: Draelos et al, IJCNN 2017
Spiking network algorithms: Severa et al., ICRC 2016
Digital neuromorphic architecture: Smith et al., IJCNN 2017

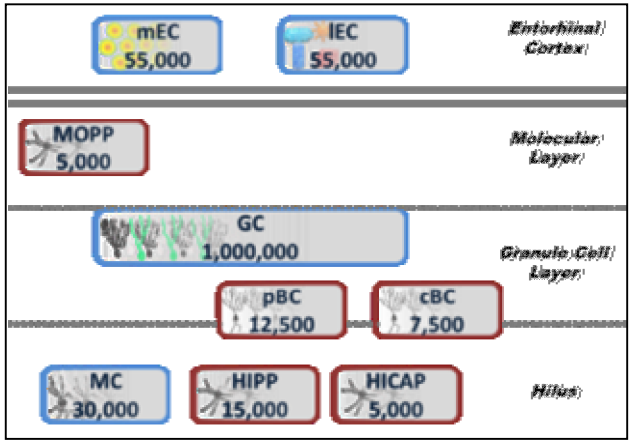
Resistive switching model: Mickel et al, Adv Mater 2014
Electrochemical transistor: Fuller et al., Adv Mater 2016
Resistive crossbar accelerator: Agarwal et al., IJCNN 2016

Translating neuroscience into the next generation of computing – Neural Machine learning (NML)

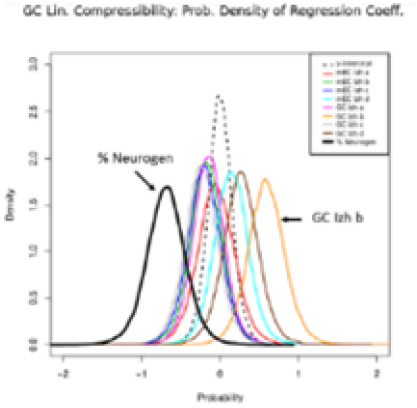
Identify neurobiological circuits of interest



Simulate at high level of neural fidelity

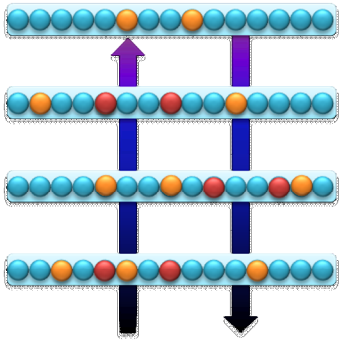


Identify critical aspects of computation

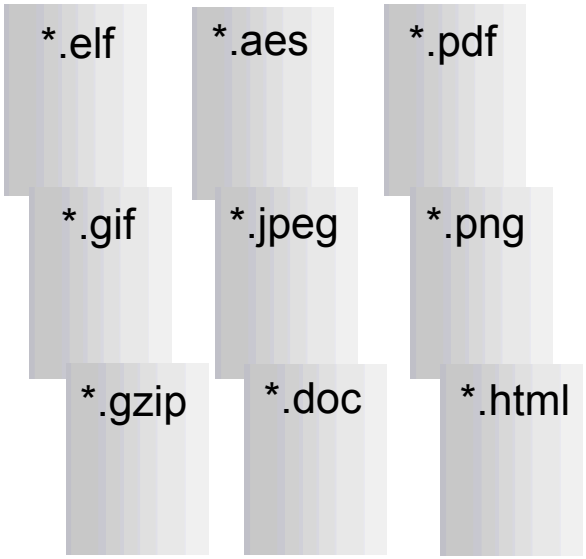


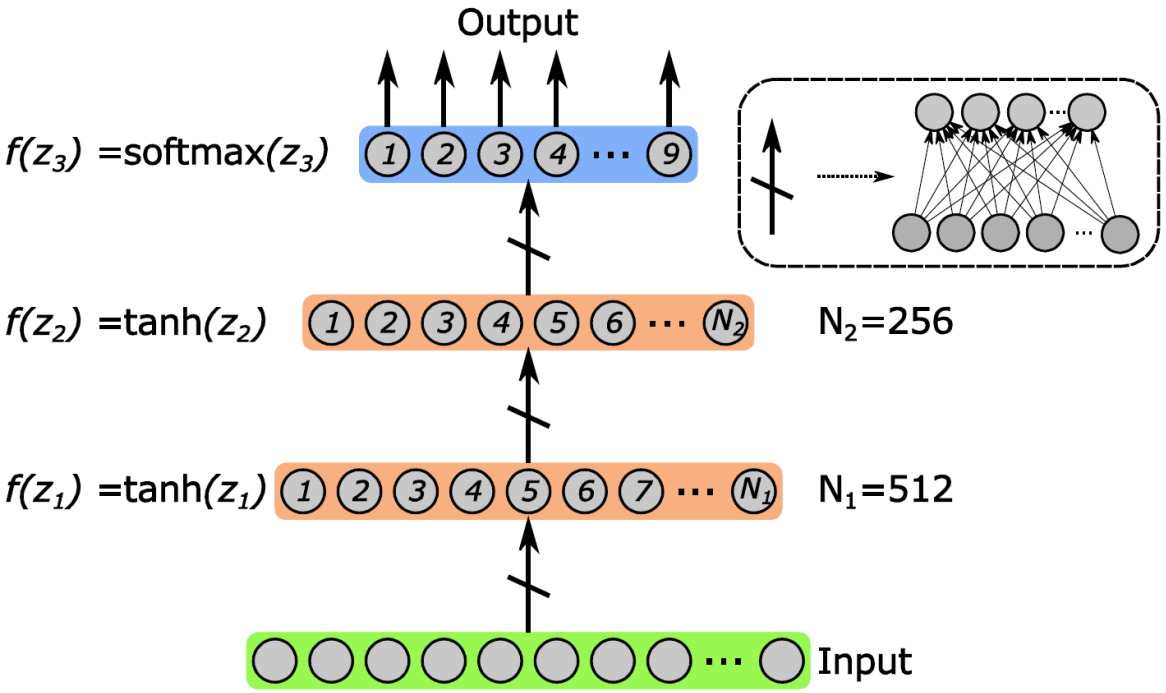
Formalize & optimize neural algorithms

Translate into NML algorithm



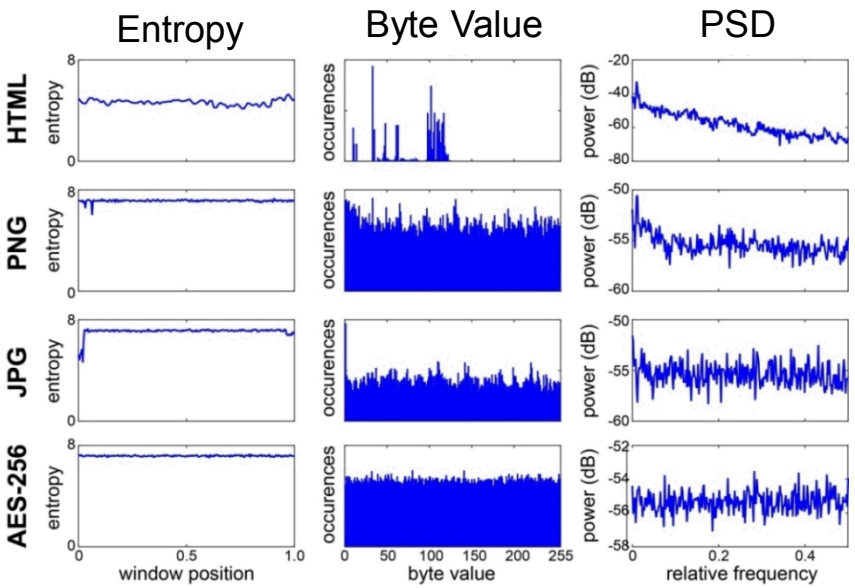
Example cyber problem: file identification using deep neural networks





Cox, Aimone, James; Complex Adaptive Systems, Nov. 2015; Procedia Comp Sci 61, 349

Limitations of supervised machine learning



3 features - Predicted Class

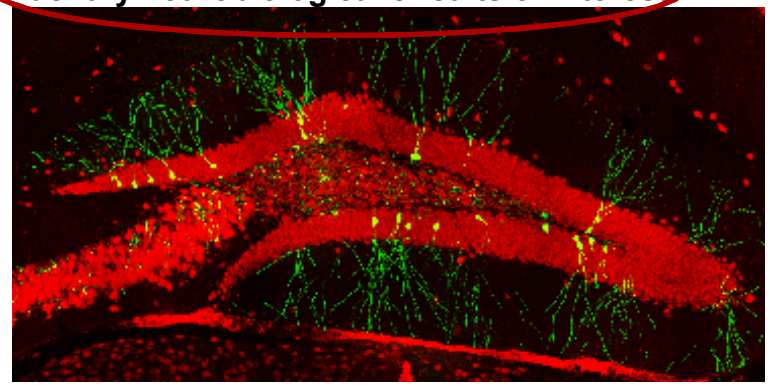
Actual Class	HTML	PNG	JPEG	GIF	PDF	DOC	ELF	GZIP	AES
HTML	100								
PNG		91	1		1	1	1	1	4
JPEG		1	99						
GIF				100					
PDF		1	3	1	95				
DOC		1				99			
ELF							100		
GZIP								100	
AES		5							95

- Supervised learning requires subject matter experts to hand-craft features
- Data-driven algorithms are limited…by the data

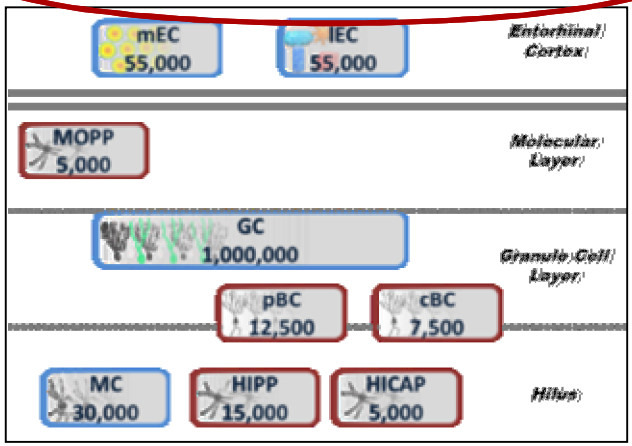
Cox, Aimone, James; Complex Adaptive Systems, Nov. 2015; Procedia Comp Sci 61, 349

Translating neuroscience into the next generation of computing

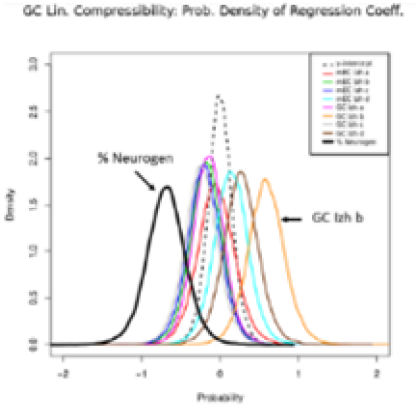
Identify neurobiological circuits of interest



Simulate at high level of neural fidelity

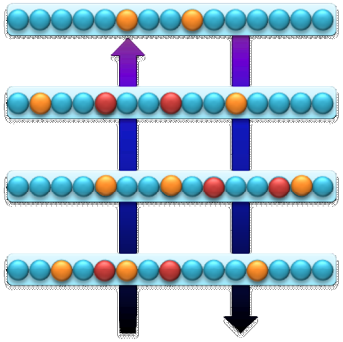


Identify critical aspects of computation

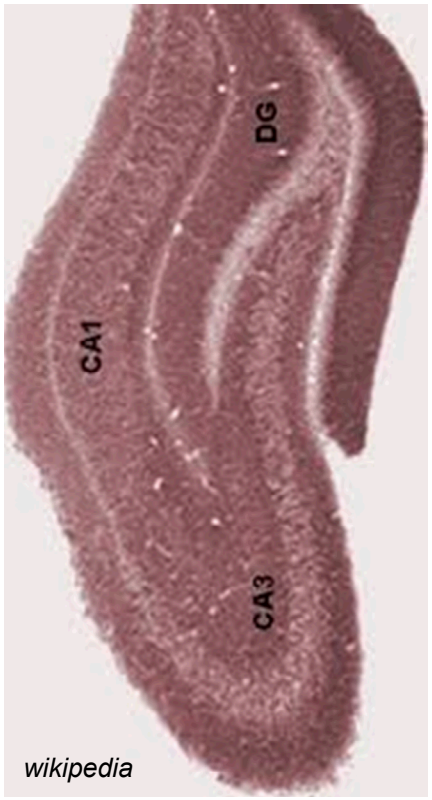


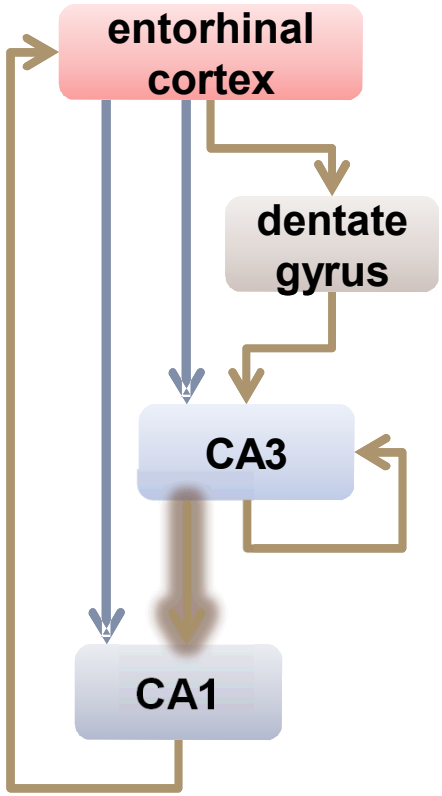
Formalize & optimize neural algorithms

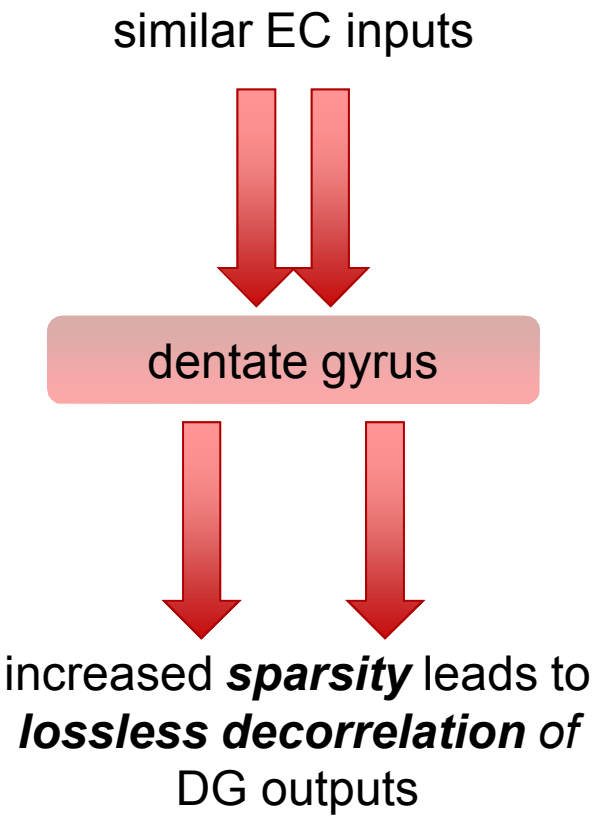
Translate into NML algorithm



Leveraging computational neuroscience models to develop new algorithms



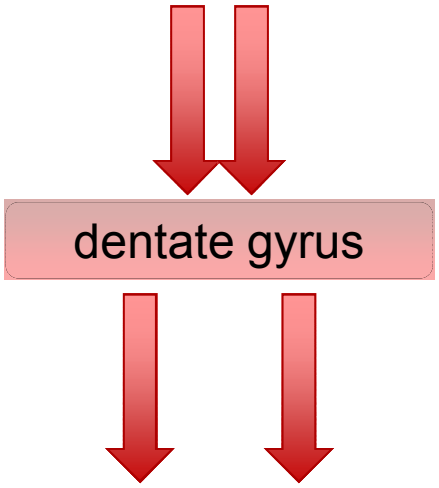




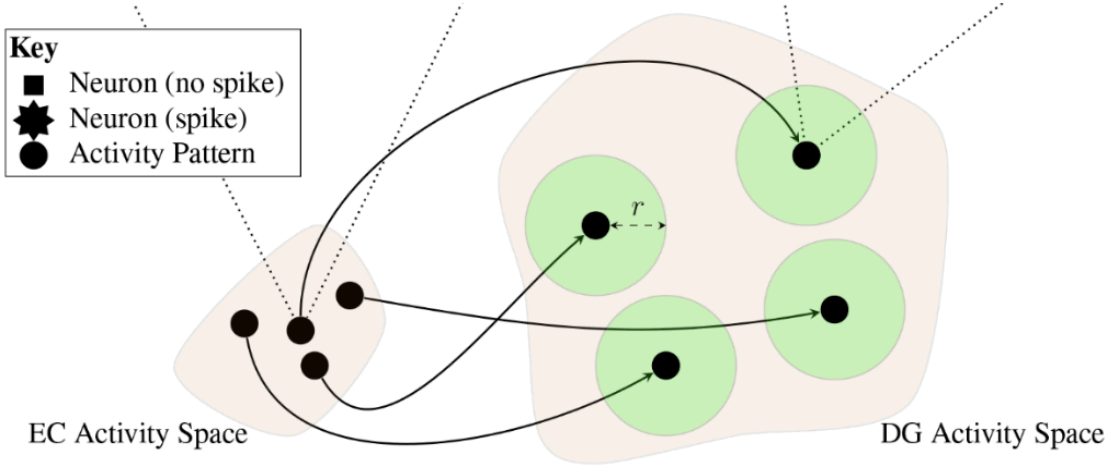
Vineyard et al., IJCNN 2016, DOI: 10.1109/IJCNN.2016.7727884

Modeling the pattern separation function of the hippocampus

similar EC inputs



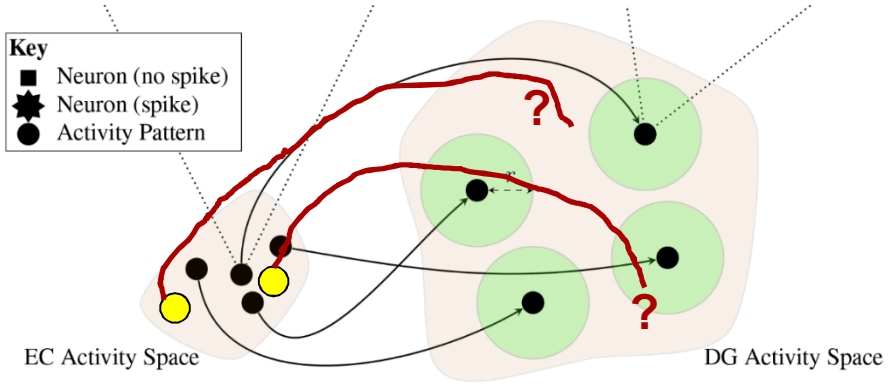
increased ***sparsity*** leads to ***decorrelated*** DG outputs

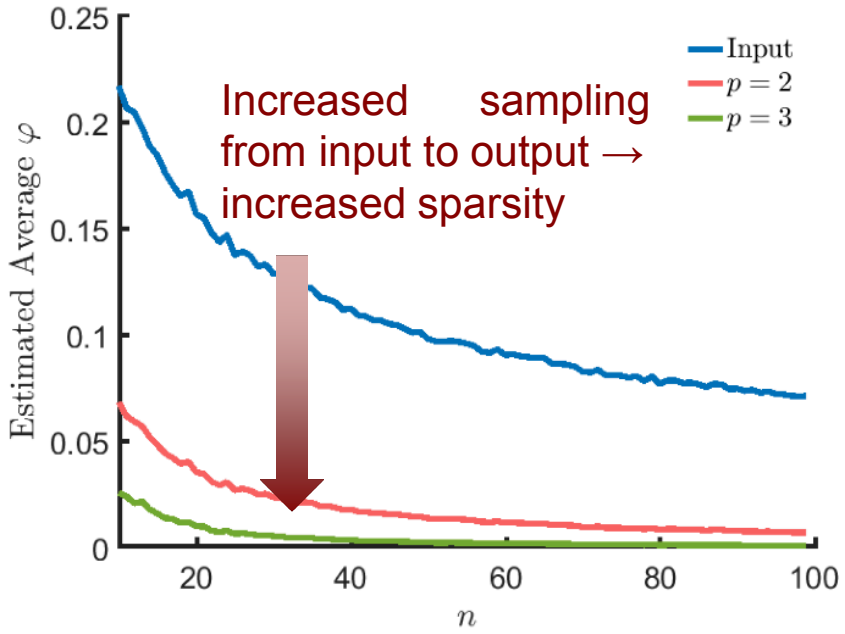


$$NDP(\mathbf{x_i}, \mathbf{x_j}) = \frac{\mathbf{x_i} \bullet \mathbf{x_j}}{\|\mathbf{x_i}\| \times \|\mathbf{x_j}\|}$$

Severa et al., Neural Computation 2017, 29, 94

Quantifying the sparsity transformation in the hippocampus

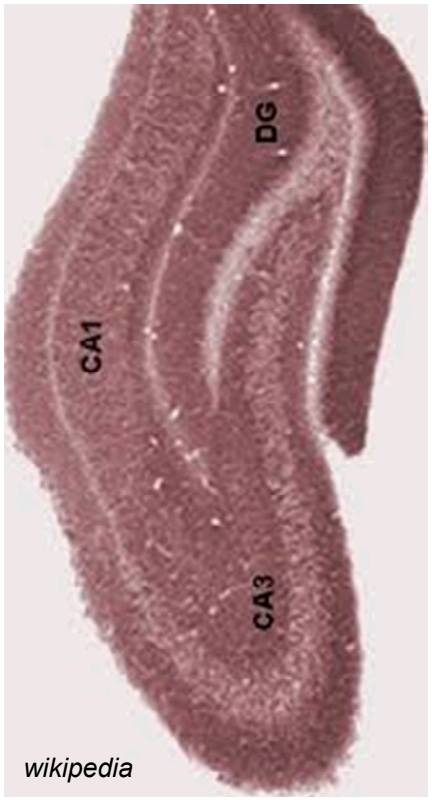


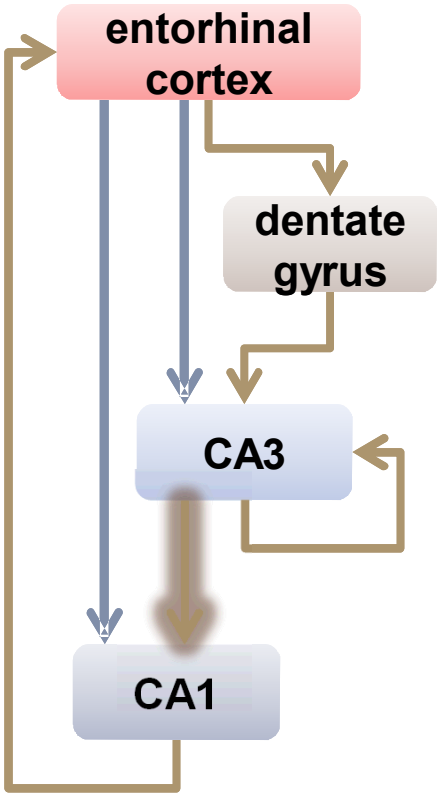


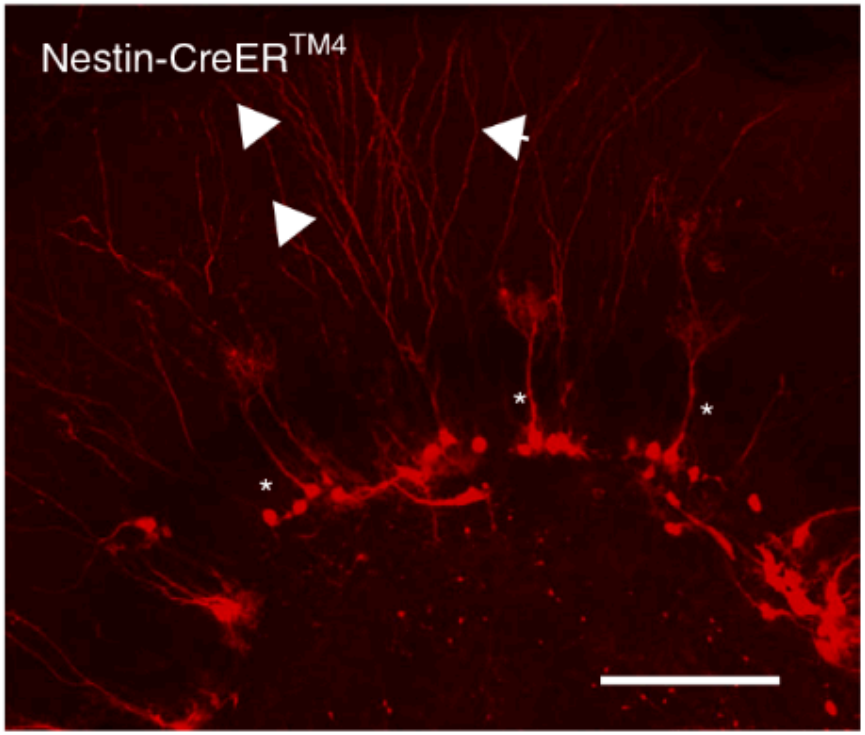
This approach is feasible for well-defined inputs – need to formalize an algorithm to account for unknown inputs

Severa et al., Neural Computation 2017, 29, 94

Algorithm inspiration: neurogenesis in biological networks



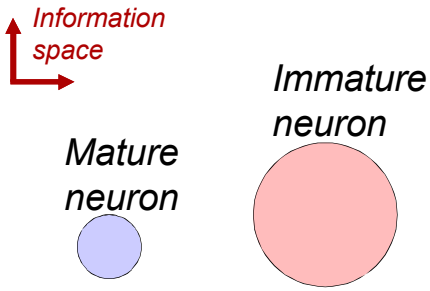




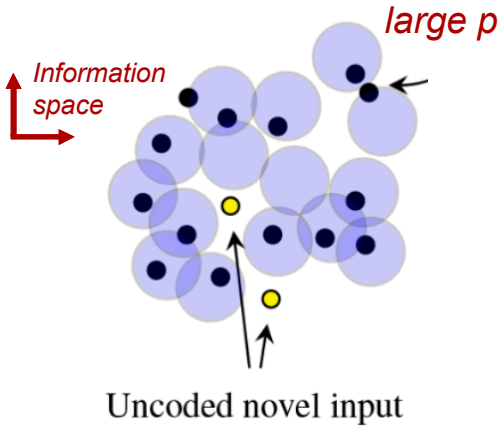
Dieni et al, Nature Comm 2016, 7:11313

Adult neurogenesis improves information capacity…?

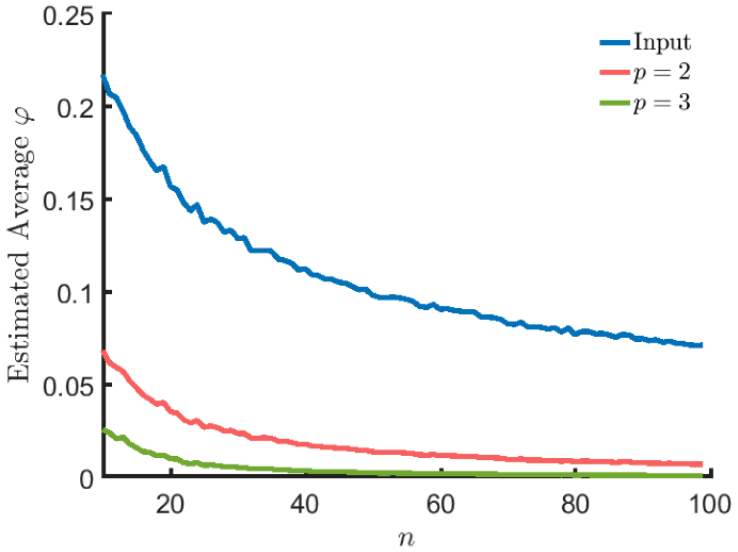
Neurogenesis may provide flexible encoding strategies for particular brain regions

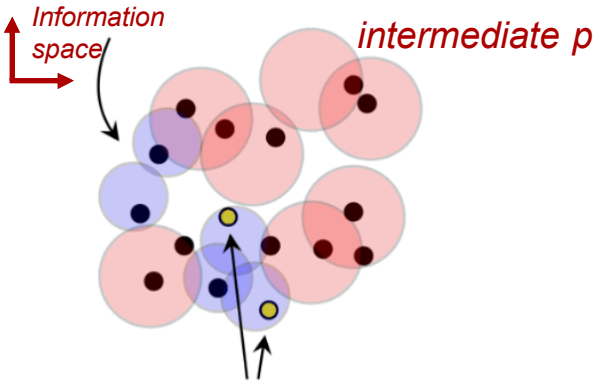


Aimone, Deng and Gage, Neuron 2011, 70, 589



Uncoded novel input



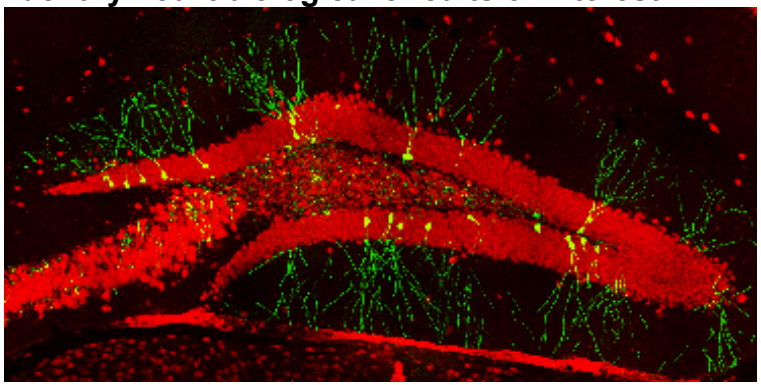


Differentiated novel input

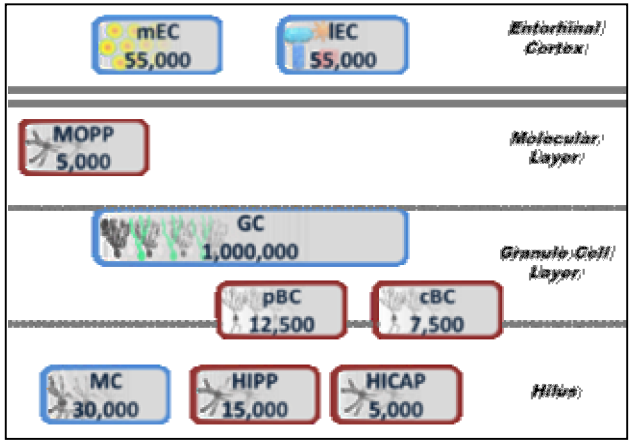
Severa et al., Neural Computation 2017, 29, 94

Translating neuroscience into the next generation of computing

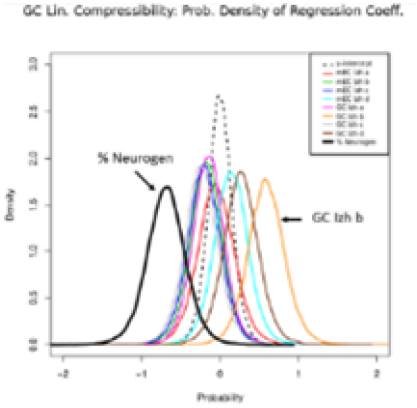
Identify neurobiological circuits of interest



Simulate at high level of neural fidelity

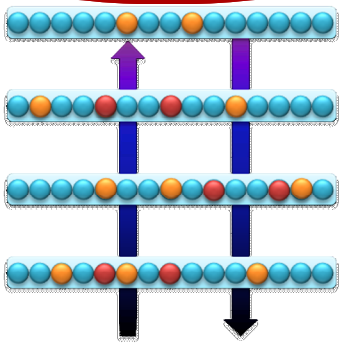


Identify critical aspects of computation

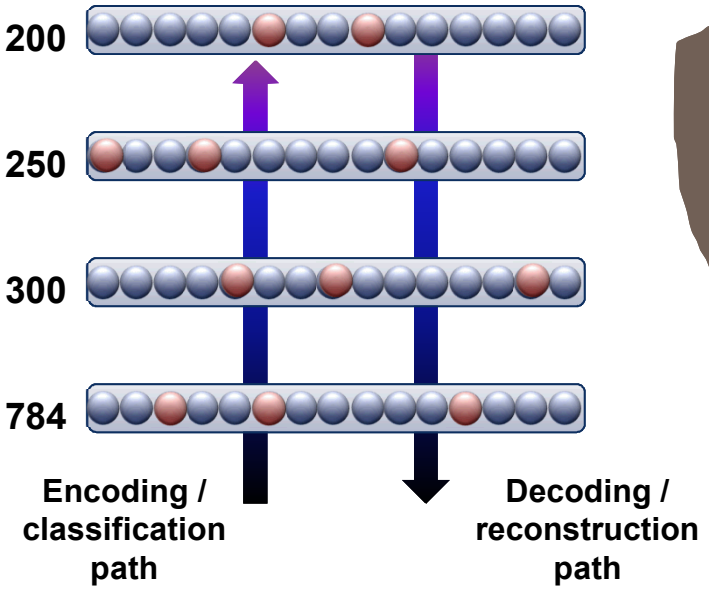


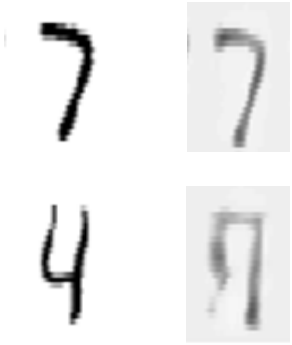
Formalize & optimize neural algorithms

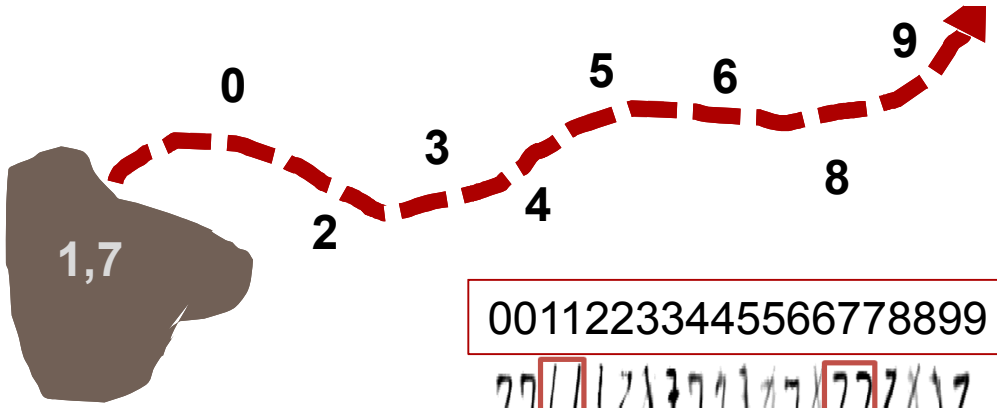
Translate into NML algorithm

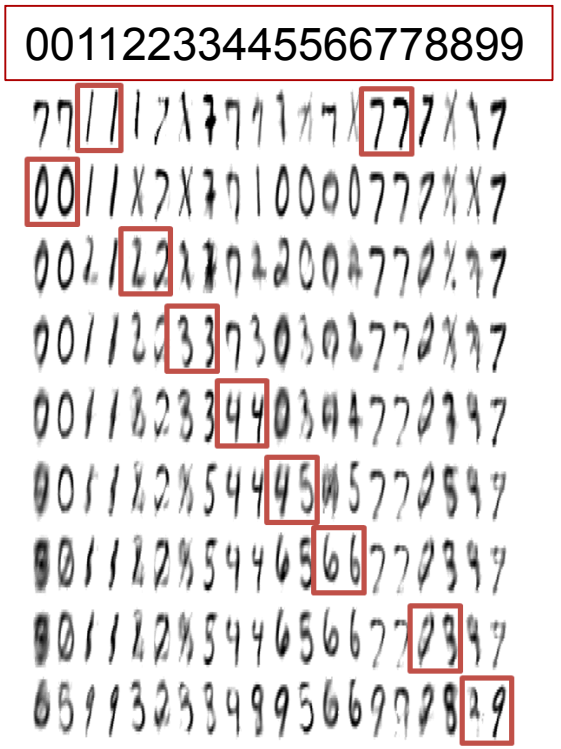


Data-driven computing methods are limited… by data



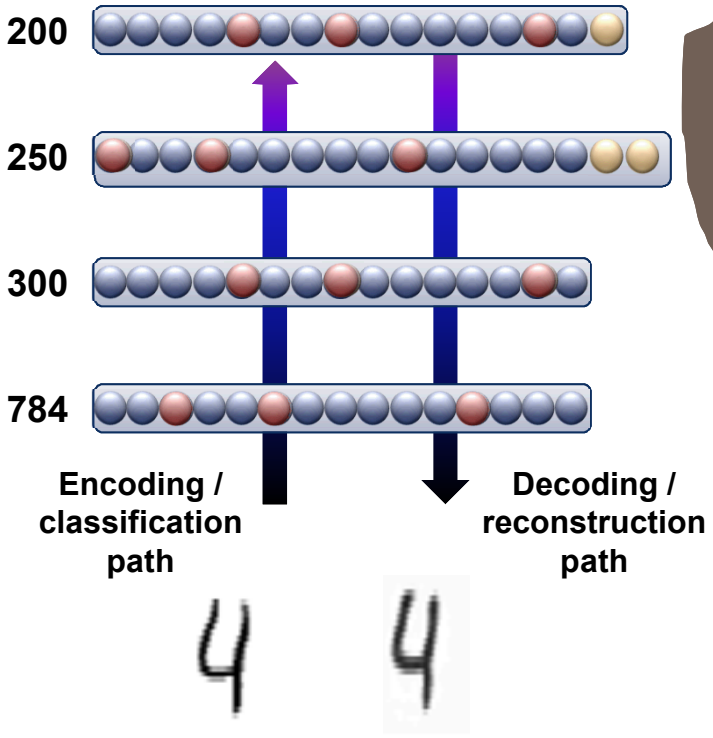


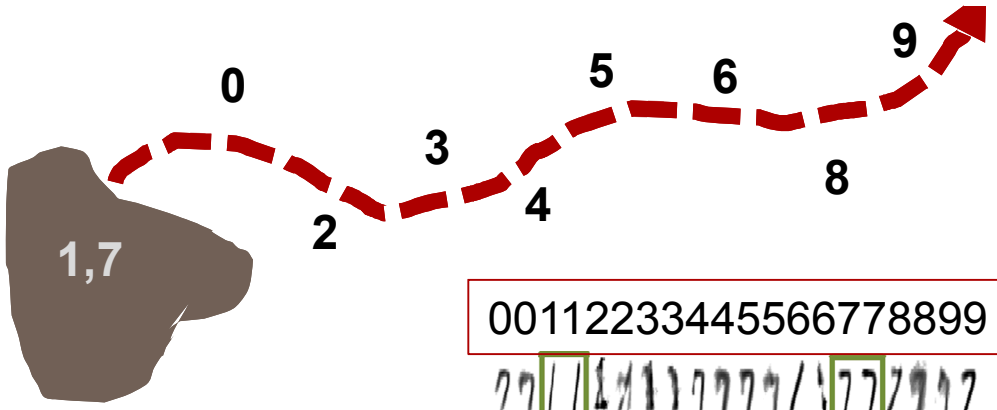




Draelos et al, ICLR 2016, IJCNN 2017

“Neurogenic deep learning” enables adaptation to changing data

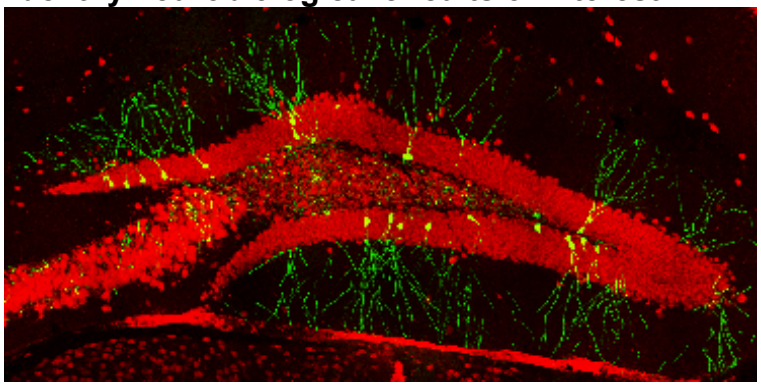




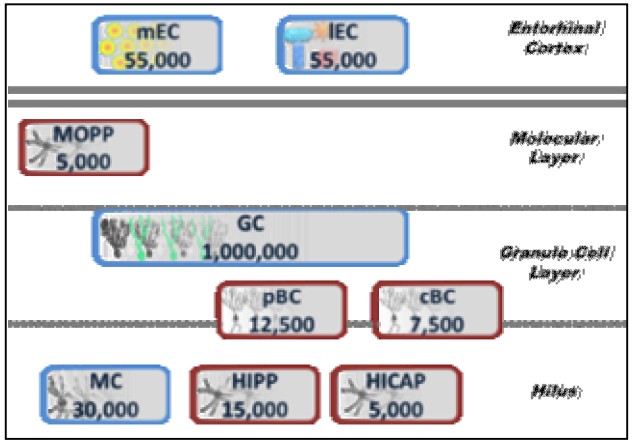
Draelos et al, ICLR 2016, IJCNN 2017

Translating neuroscience into the next generation of computing

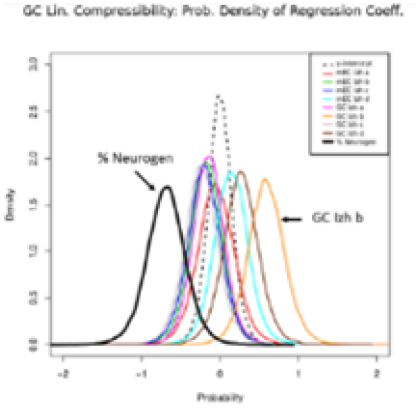
Identify neurobiological circuits of interest



Simulate at high level of neural fidelity

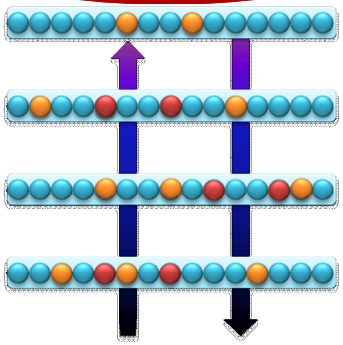


Identify critical aspects of computation

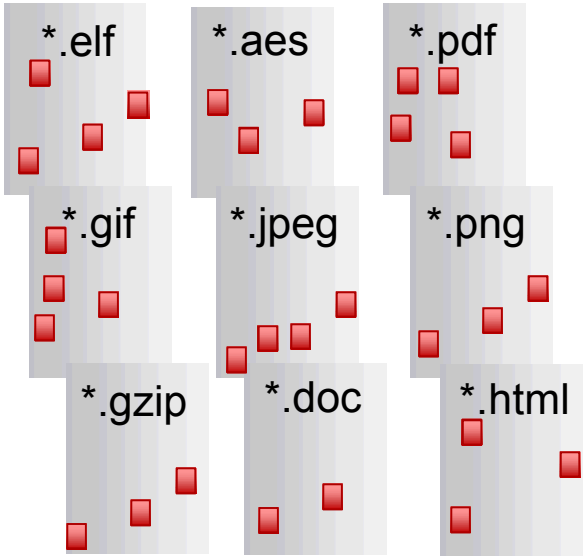


Formalize & optimize neural algorithms

Translate into NML algorithm

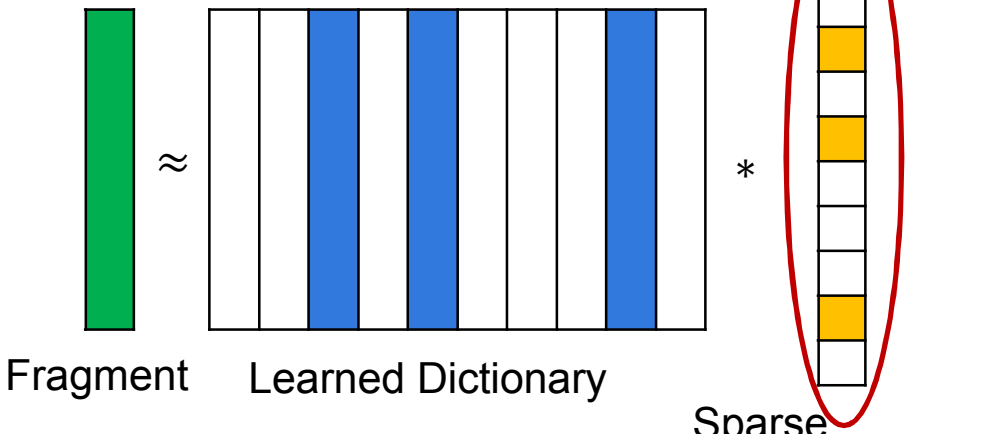


Categorizing cyber data under imperfect conditions with sparse coding



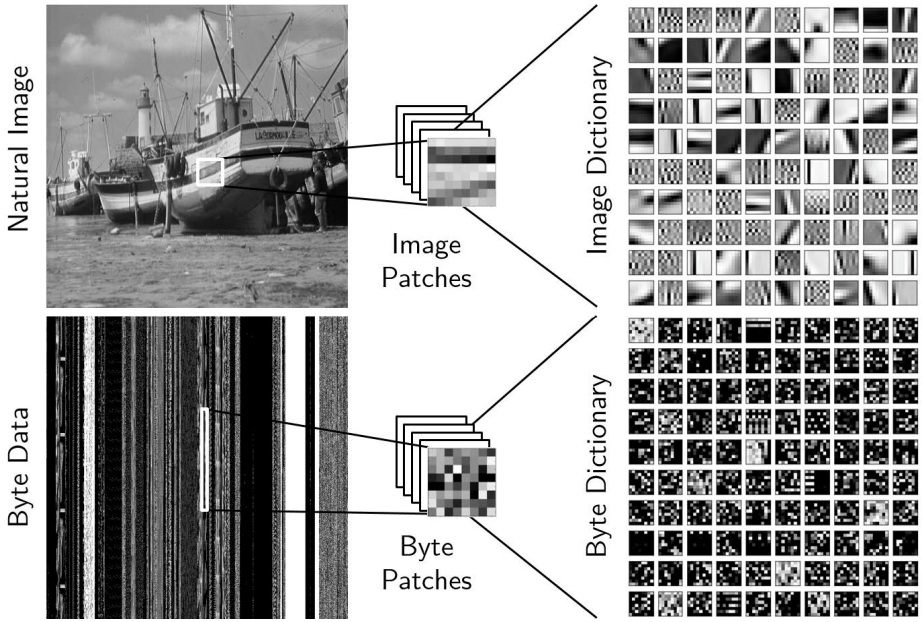
$$y \approx Ax$$

$$\min_x \|y - Ax\|^2 + S(x)$$

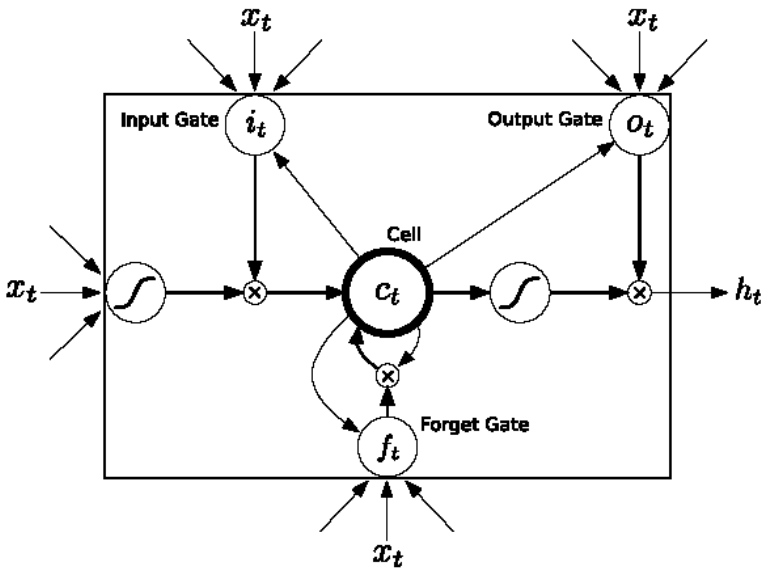


- Training data is not always available, fragmented
- Limited expertise; hand-engineered features

Generating local and global features from file fragments



• Byte dictionary patches & sparse representations of fragments
– local features



Graves et al., ASRU2013
Hochreiter & Schmidhuber, Neur Comp 1997

• Long short-term memory (LSTM) networks are used to improve long-range correlations
– global features

Wang et al., in preparation

Sparse dictionary learning and LSTM networks for file fragment ID – compared to SVM

Predicted class

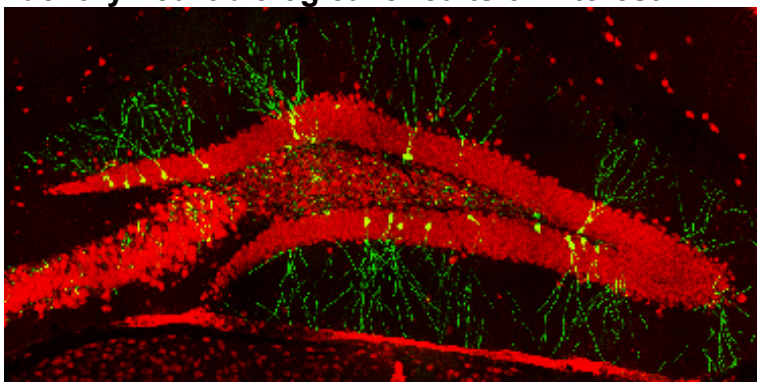
Actual class	csv	doc	gif	gz	html	jpg	pdf	png	ppt	ps	rtf	swf	txt	xls	xml
csv	-9.8	-0.9			-3.8		-0.3			-0.3	-1.1		-0.9		-2.5
doc	-0.5	11.9	-0.5	6.6	-3.0	-1.0	-0.4	4.1	-1.5	-0.5	-3.6	-1.0	14.0	0.6	-1.4
gif			36.5	29.3		-10.0		17.3				-0.1			
gz			-12.0	-28.1		-31.8	-0.4	16.6				-0.5			
html	-1.4	-9.0			-17.5		-0.4		-0.6	-1.6	-10.4		-3.4	-0.1	9.4
jpg		0.4	-9.6	18.7		15.2	-1.7	6.4	-0.4			0.2	1.2	-0.4	0.4
pdf		-0.1	-7.2	15.8	-0.6	-17.9	9.4	12.6	-1.2	5.7	-0.5	-0.3	3.0	-0.1	0.2
png			-13.4	9.1		-21.0	-0.4	-26.7	-0.2			-0.8			
ppt		3.2	-5.7	15.0	-0.1	-14.3	-0.1	11.2	9.1	0.2	-0.1	1.0	0.7	-2.1	0.2
ps	-1.4		-0.8	0.2	-2.3	-1.1	-0.3	-0.3		-3.2	-1.1		5.7	-1.8	
rtf	-0.5	-6.2	0.6		-6.1	0.4	-0.7			3.0	-4.2		6.3		-1.0
swf		0.7	-5.3	19.8		-23.6	-1.0	10.2	-0.9	-0.1		2.0		1.8	0.4
txt	-4.3	-4.2			-8.2		0.2			9.3	-4.8		-15.4	-0.6	-2.8
xls		11.7	-0.3	0.2	-2.2	-0.2	-0.4	0.8	-0.8	-0.3	-0.2	-0.8	-0.9	6.3	-0.3
xml	-0.6	-2.0			16.1		1.2			-0.6	-4.2		6.0		15.9

- Averaged F_1 score = 53.12%
- *NLP (SVM) approach achieved 49.1 ± 3.15% (Fitzgerald et al., DI 2012)*

Wang et al., in preparation

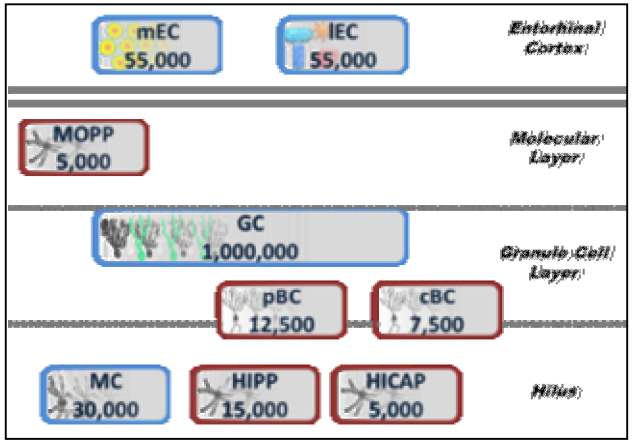
Translating neuroscience into the next generation of computing

Identify neurobiological circuits of interest

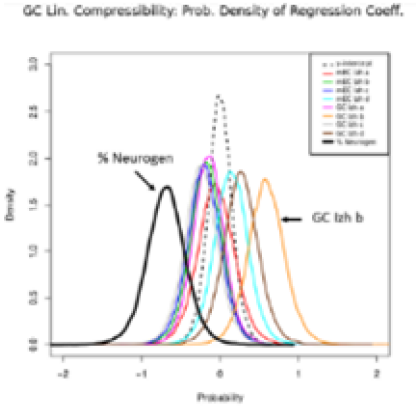


Formalize & optimize neural algorithms

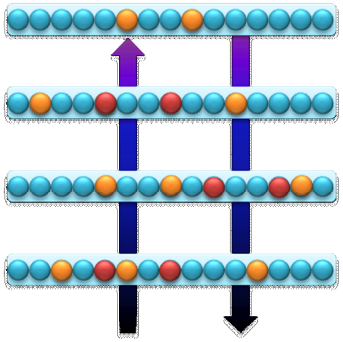
Simulate at high level of neural fidelity



Identify critical aspects of computation



Translate into NML algorithm



Optimization of algorithm performance

Neural algorithm operations are computationally expensive (energy and time) due to training; many matrix-vector operations

Sparse coding:

$$\min_x \|y - Ax\|^2 + S(x)$$

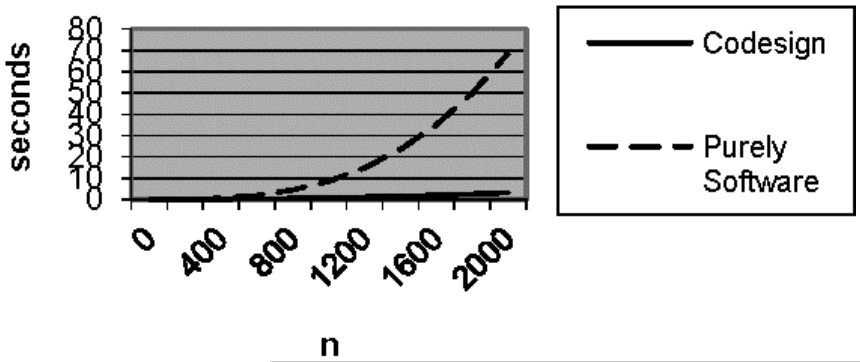
Backpropagation: $\Delta_k = \sum_k w_{jk}\delta_k$

Make better/smarter algorithms:

$$f * g = F^{-1}\{F\{f\} \cdot F\{g\}\}$$

$$|j\rangle \mapsto \frac{1}{\sqrt{N}} \sum_{k=0}^{N-1} \omega^{jk} |k\rangle$$

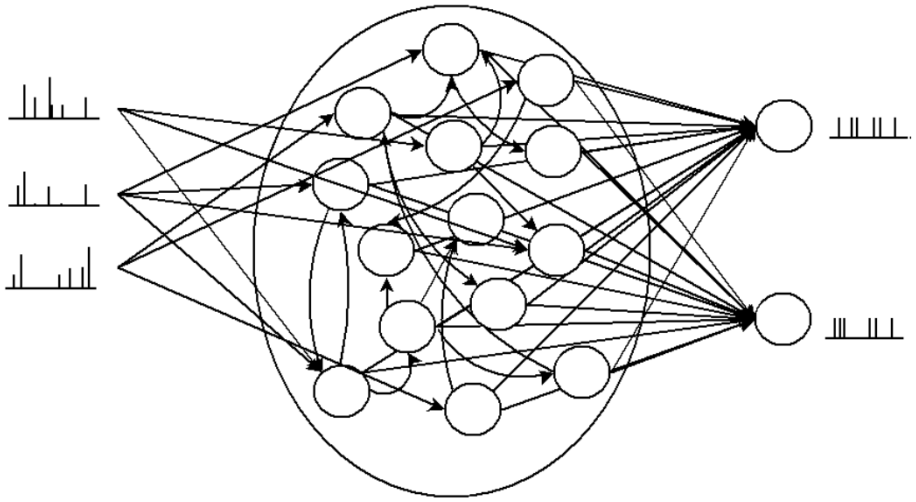
Hardware accelerate algorithms:



Lee et al., Proc World Cong Eng Comp Sci 2013

Hardware acceleration of spiking algorithms for time-dependent data processing

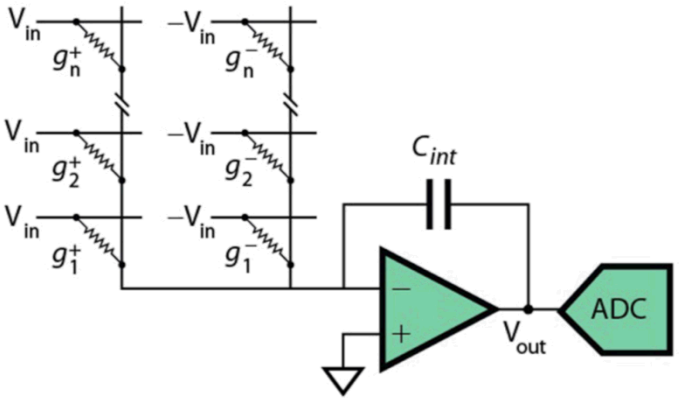
Example: liquid state machine(LSM); a tool for data transformation; randomly connected spiking neurons encode complex ***temporal dynamics***



Spiking algorithms are often inefficient on conventional hardware…

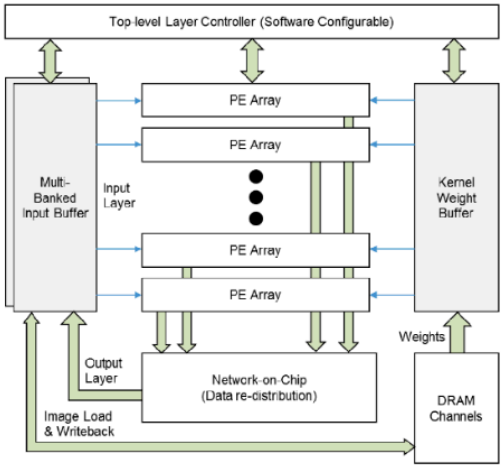
Maass et al, Neural Comp 2002, 14, 2531

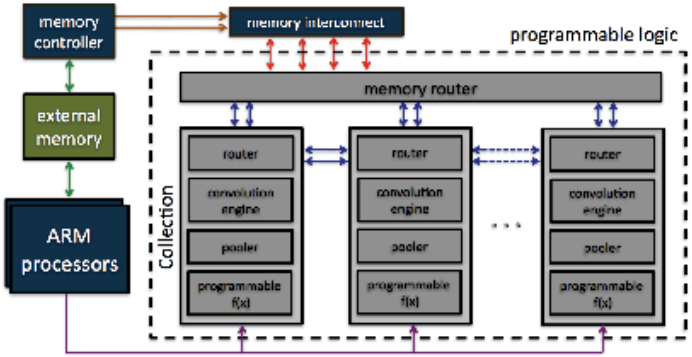
Hardware acceleration of algorithm operations



Gokmen & Vlasov., (IBM), Resistive crossbar acceleration of DNNs, 2016

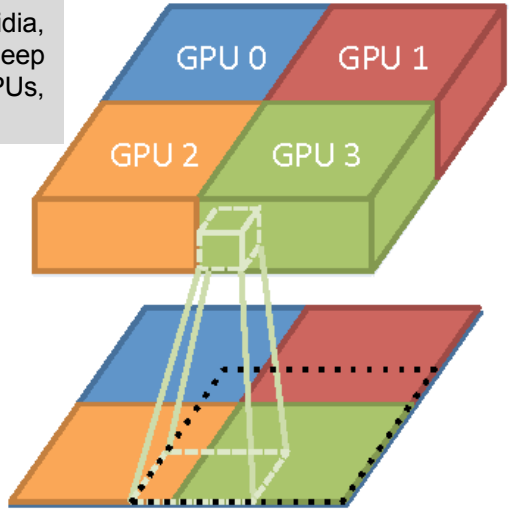
Ovtcharov et al., (Microsoft), FPGA acceleration of CNNs, 2015





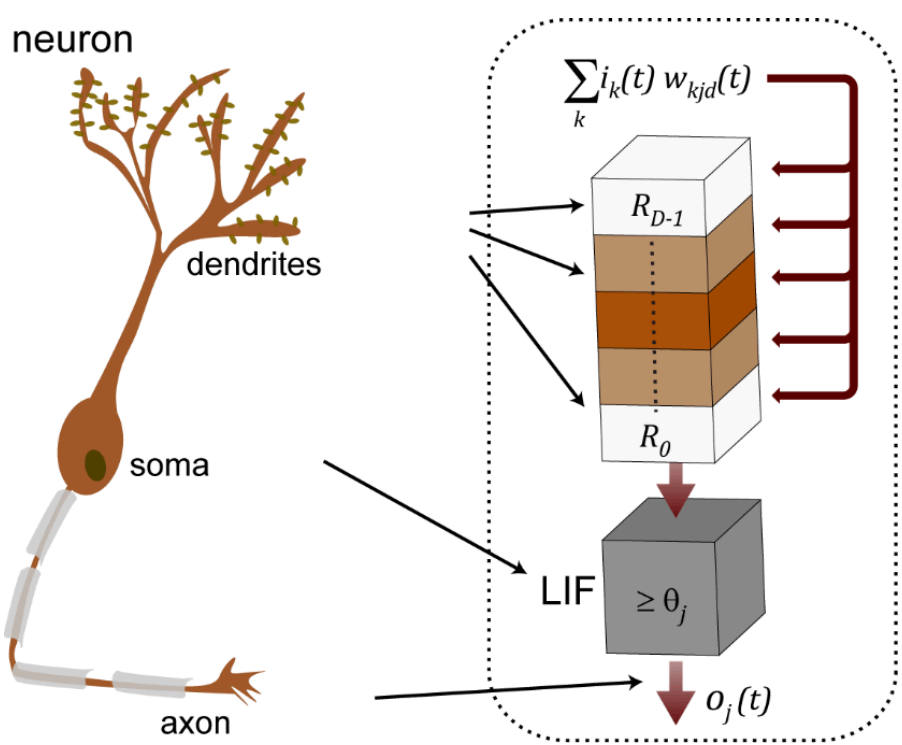
Gokhale et al., (Purdue), nnX for accelerating DNNs with ARMs, 2013

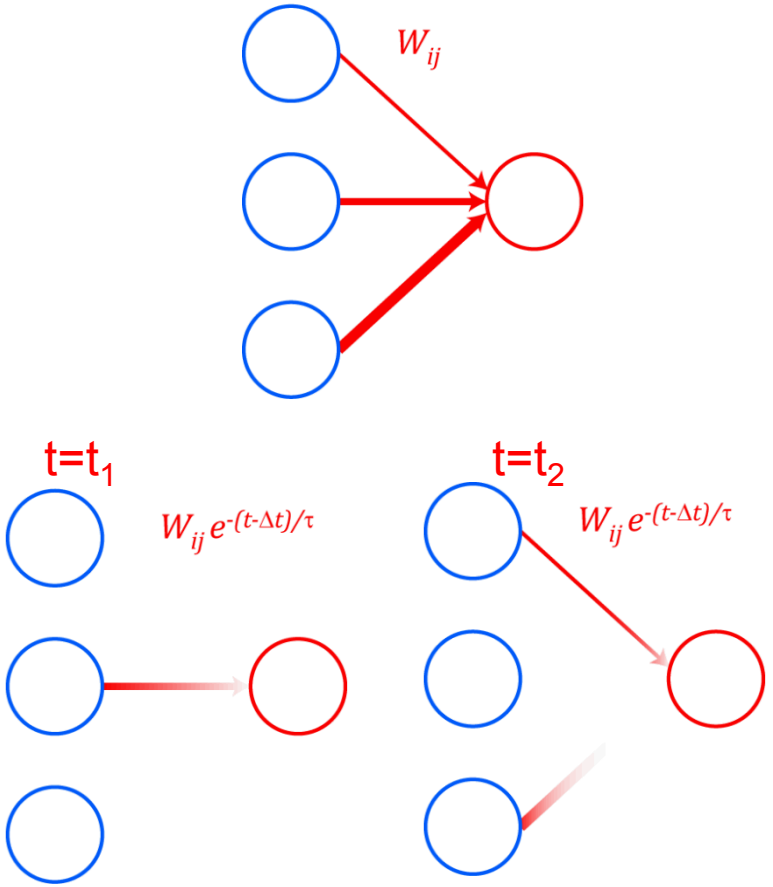
Coates et al., (Nvidia, Stanford), Deep learning with GPUs, 2013



Spiking Temporal Processing Unit (STPU)

Impart complex temporal dynamics into neural networks

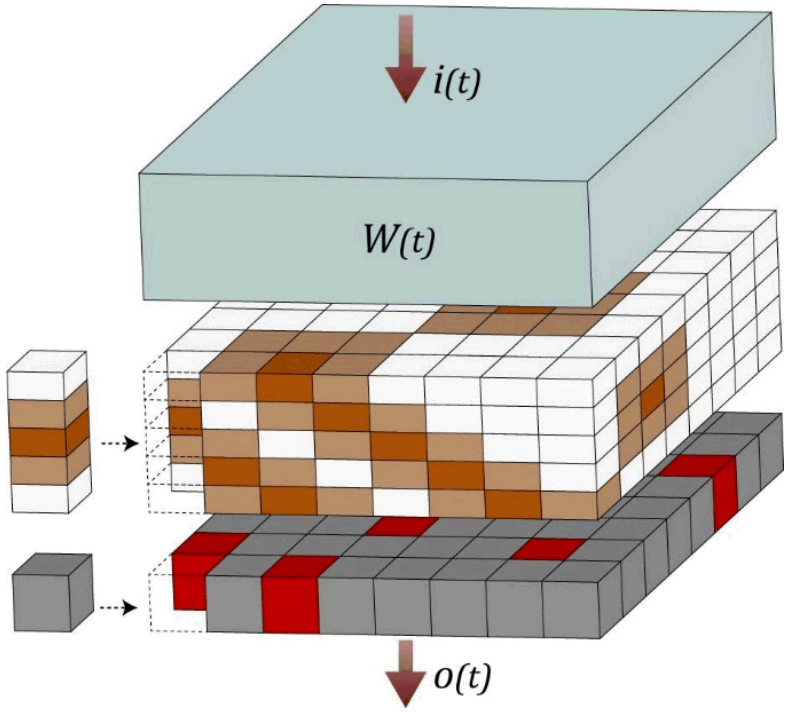




Smith et al, "A Novel Digital Neuromorphic Architecture…", IJCNN 2017

Emulation of a LSM mapped onto an STPU architecture

Assemble an array of LIF neurons and combine with a synaptic map *W*

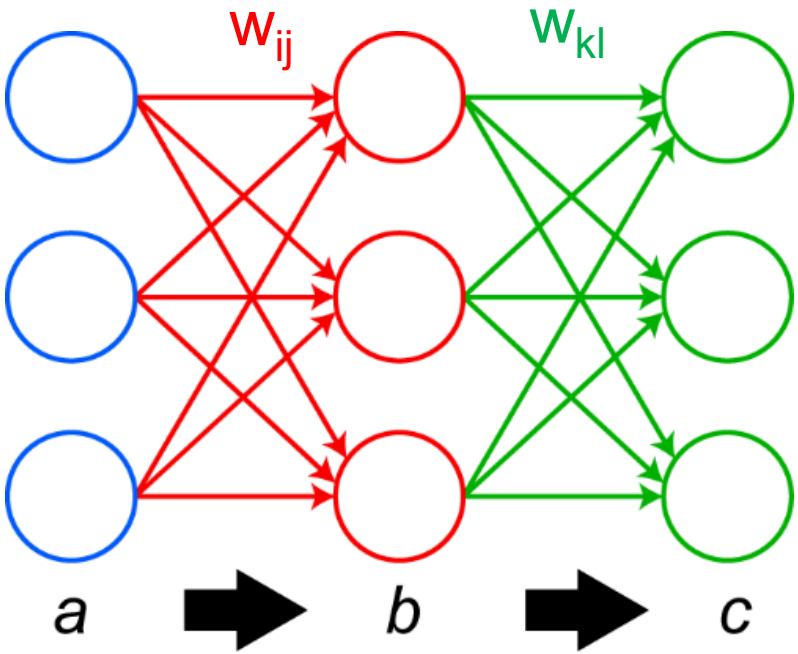


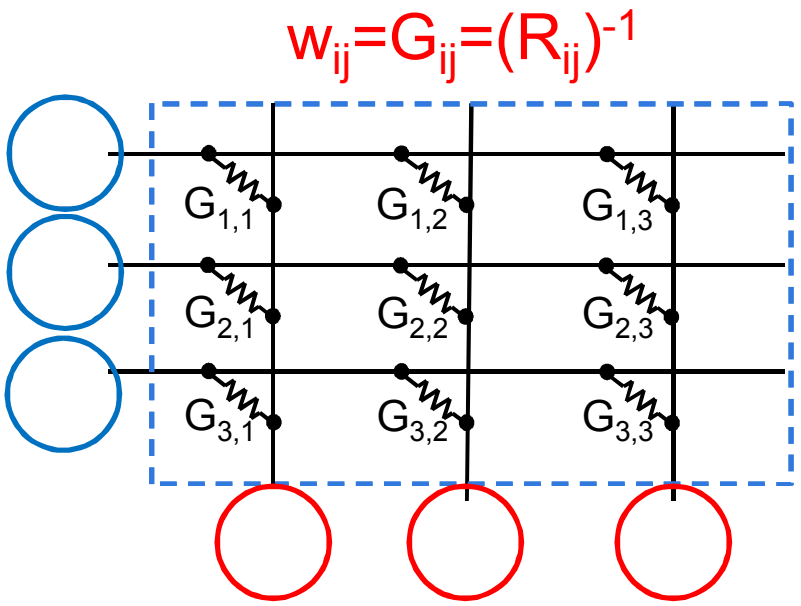
- Test data: spoken digits (0-9)
- Implement the liquid on the STPU
- Use a classifier to categorize the spoken digits

Linear Model	3x3x15	5x5x5	4x5x10	2x2x20
Linear SVM	0.906	0.900	0.900	0.914
LDA	**0.921**	**0.922**	**0.922**	**0.946**
Ridge Regress	0.745	0.717	0.717	0.897
Logistic Regress	0.431	0.254	0.254	0.815

Smith et al, "A Novel Digital Neuromorphic Architecture…", IJCNN 2017

Implementing synaptic connections in hardware for non-spiking neural algorithms



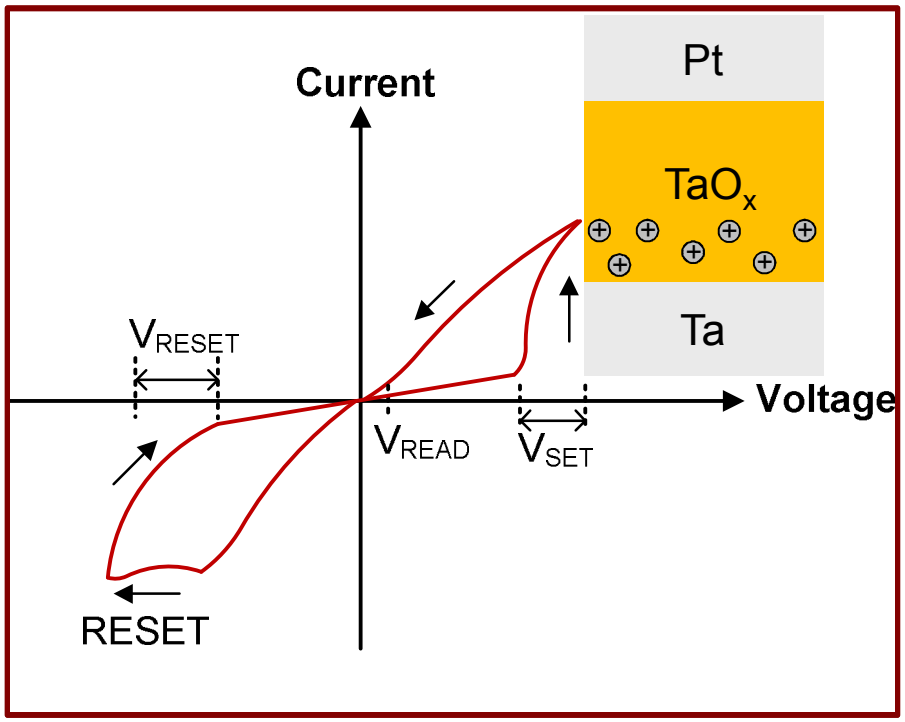


Use variable resistors to implement neural network weights in hardware – saves energy $O(n^3)$ to $O(n^2)$

Agarwal et al., Front. Neuroscience 2016, 9, 484

Agarwal et al, IJCNN 2016, DOI: 10.1109/IJCNN.2016.7727298

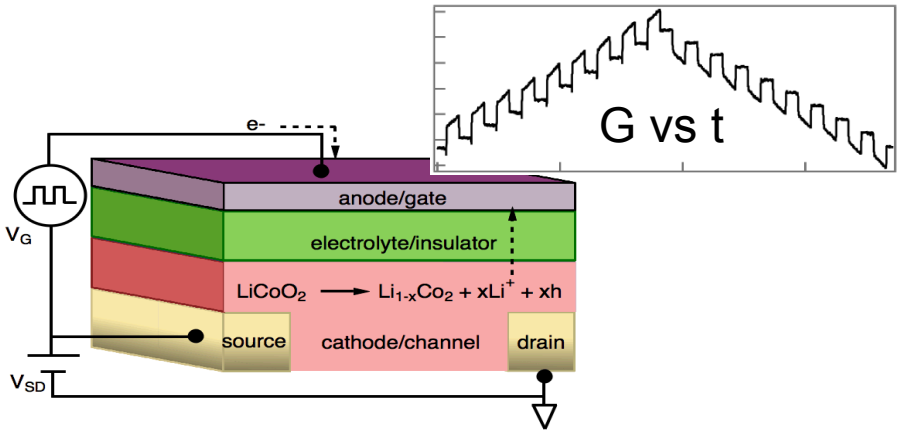
Designing, modeling, and fabricating devices with improved neural computing characteristics

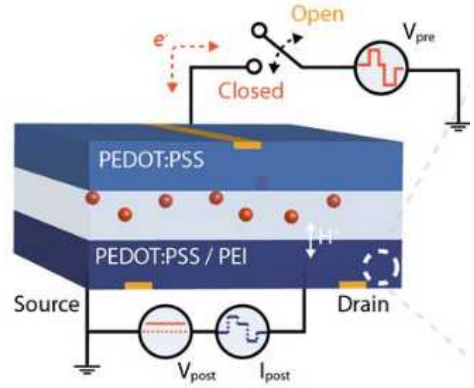


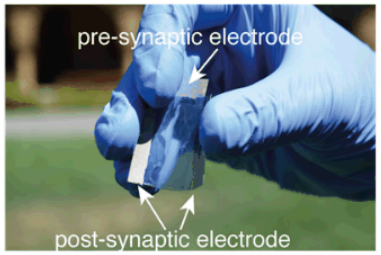
Filament surface temperature (T_s):

$$T_s = T_{RT} + \sigma V^2 \frac{d_E}{2k_E d_o}\left[1 - \frac{k_E}{k_F}\frac{r_F^2}{4d_E d_o}\right]$$

Mickel et al., Adv Mater, 26, 4486, 2014
Landon et al., APL 2015, 107, 023108

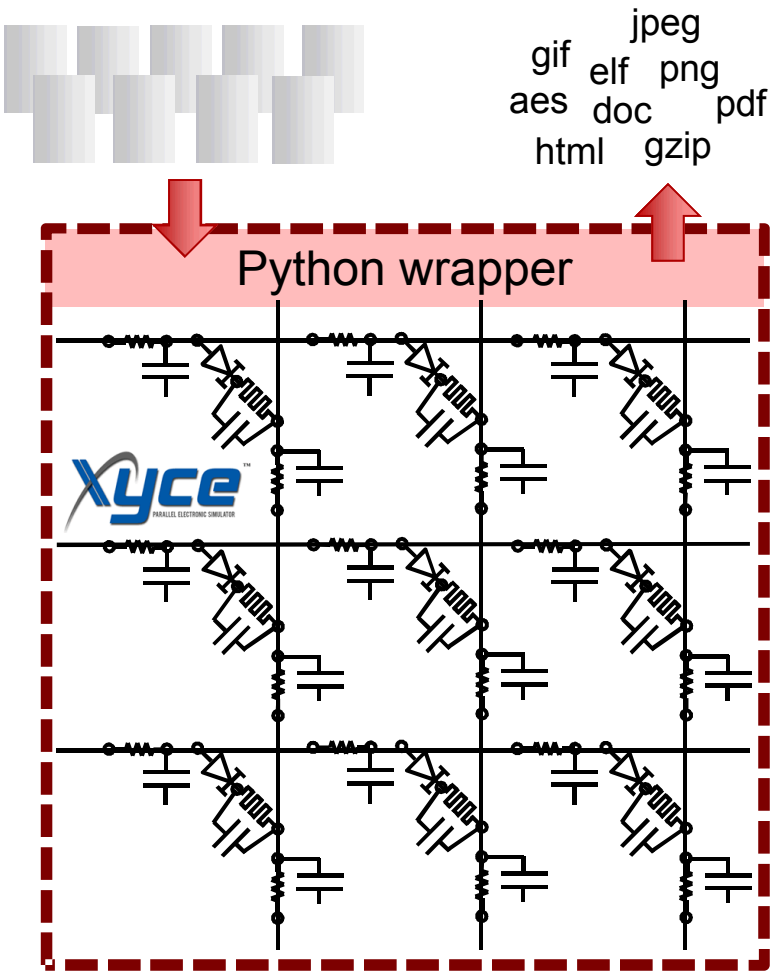


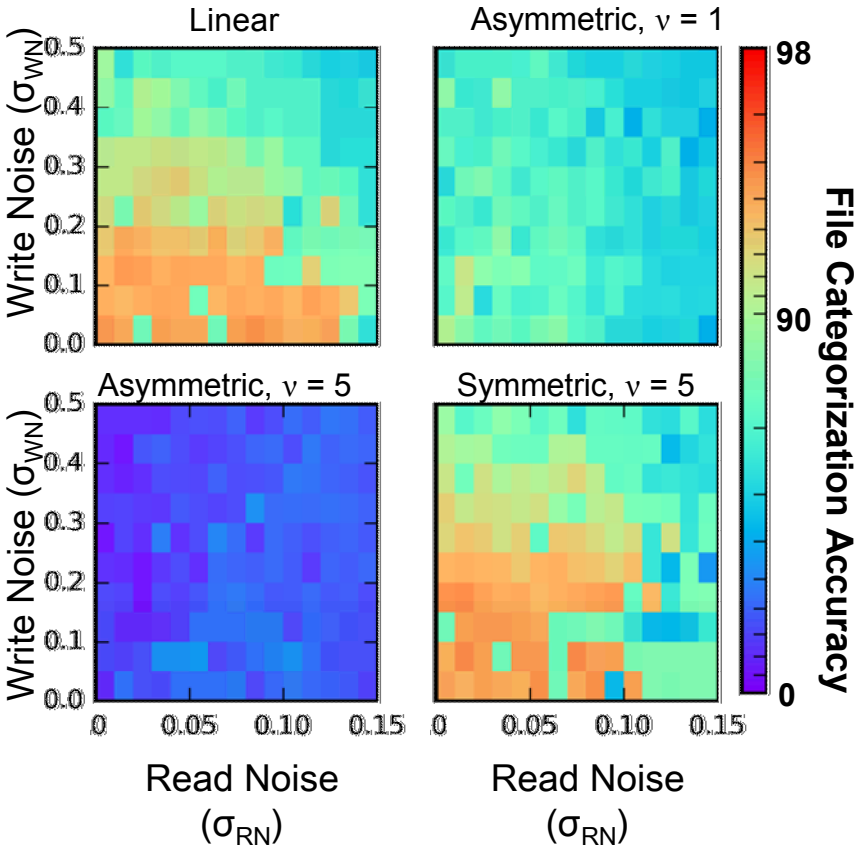




Fuller et al., Adv Mater 2016, 10.1002/adma.201604310
van de Brugt et al., Nat Materials 2017, 10.1038/nmat4856

Model hardware-acceleration to assess the impact on algorithm performance

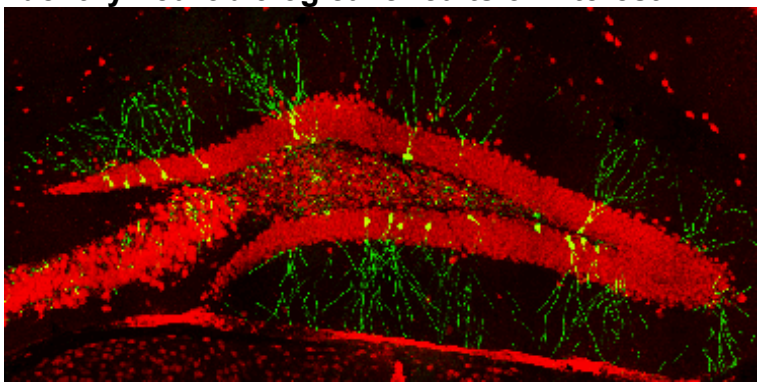




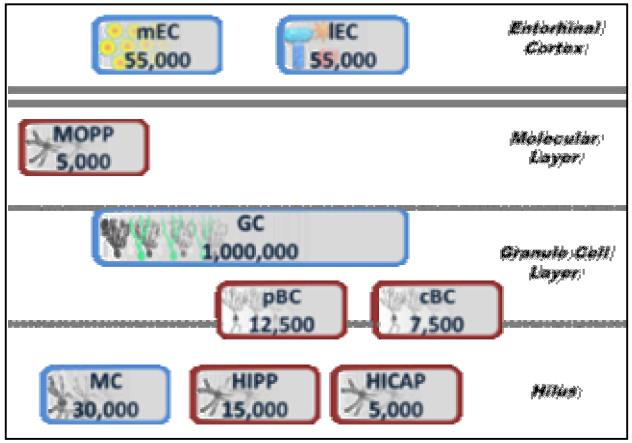
Agarwal et al, IJCNN 2016, DOI: 10.1109/IJCNN.2016.7727298

Translating neuroscience into the next generation of computing

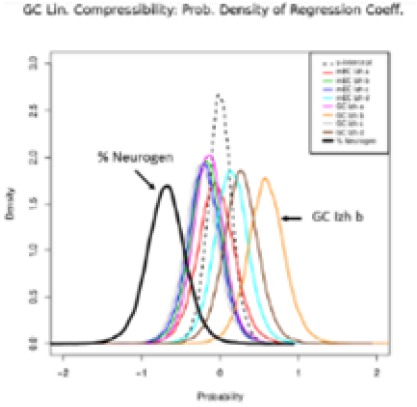
Identify neurobiological circuits of interest



Simulate at high level of neural fidelity

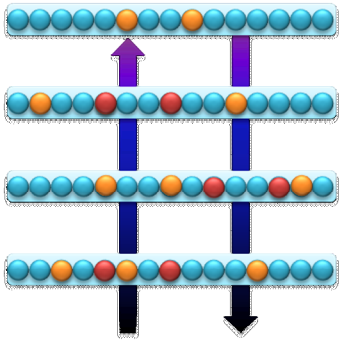


Identify critical aspects of computation



Formalize & optimize neural algorithms

Translate into NML algorithm



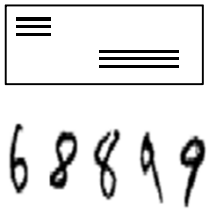
Thanks for your time!

Questions?

Backup Slides

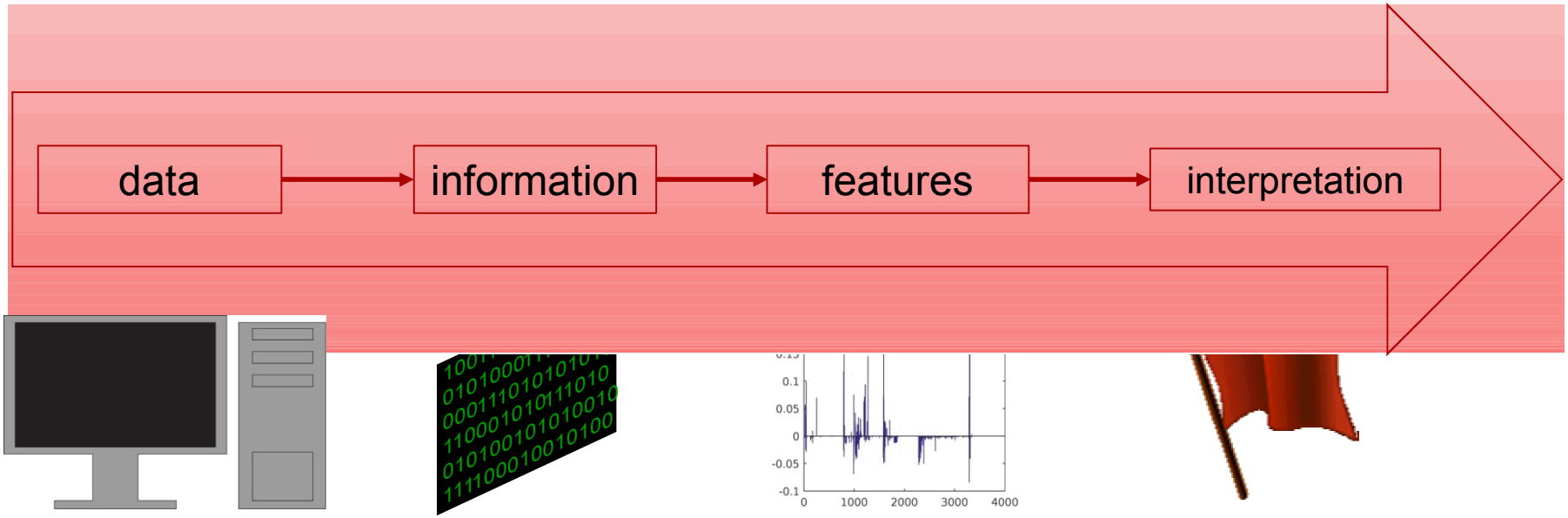
Applications in imaging and cybersecurity

- The real world is filled with massive amounts of data
- Data needs to be filtered to capture relevant information
- Signatures or features need to be extracted from data
- Features can then be used to interpret activities



‘68899’

‘zipcode’

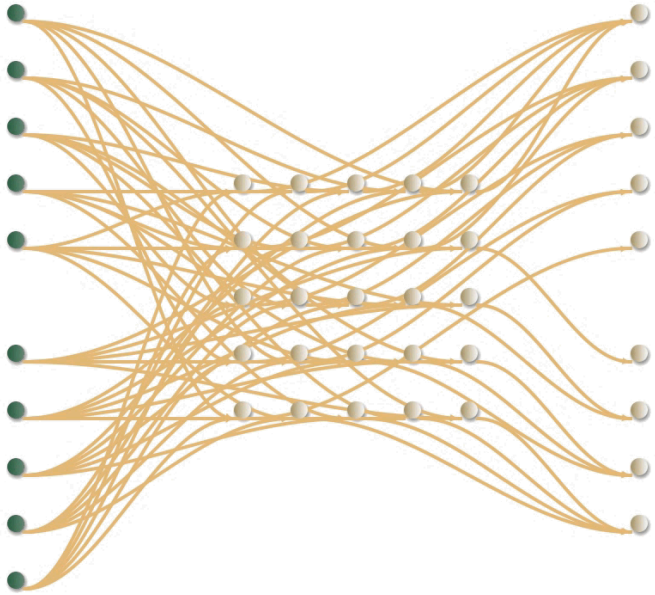


Spiking network algorithm for computing cross-correlations

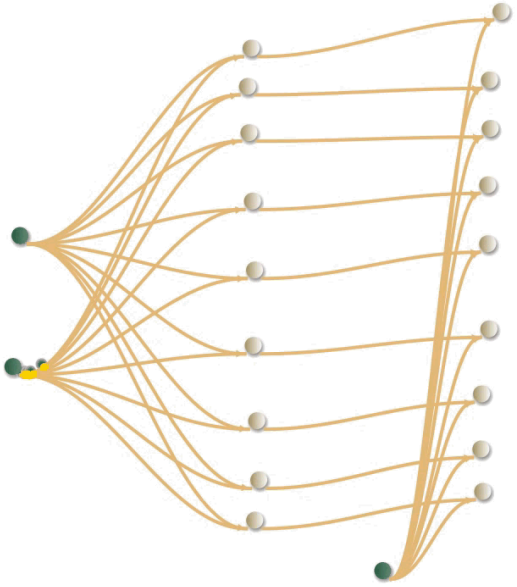
- Motivation: determine the local velocity in a flow field
- SNN algorithms are highly parallel and can leverage the time/neuron tradeoff
- Neural algorithms can match or best traditional ‘big O’

Severa et al, ICRC 2016; DOI: 10.1109/ICRC.2016.7738681

Trading neurons for time and vice versa



- $O(n^2)$ neurons, constant time



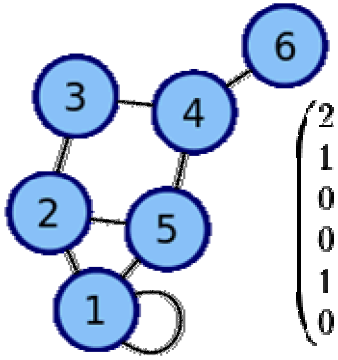
- O(n) neurons, O(n) time

Matrix operations are at the core of many neural computing operations

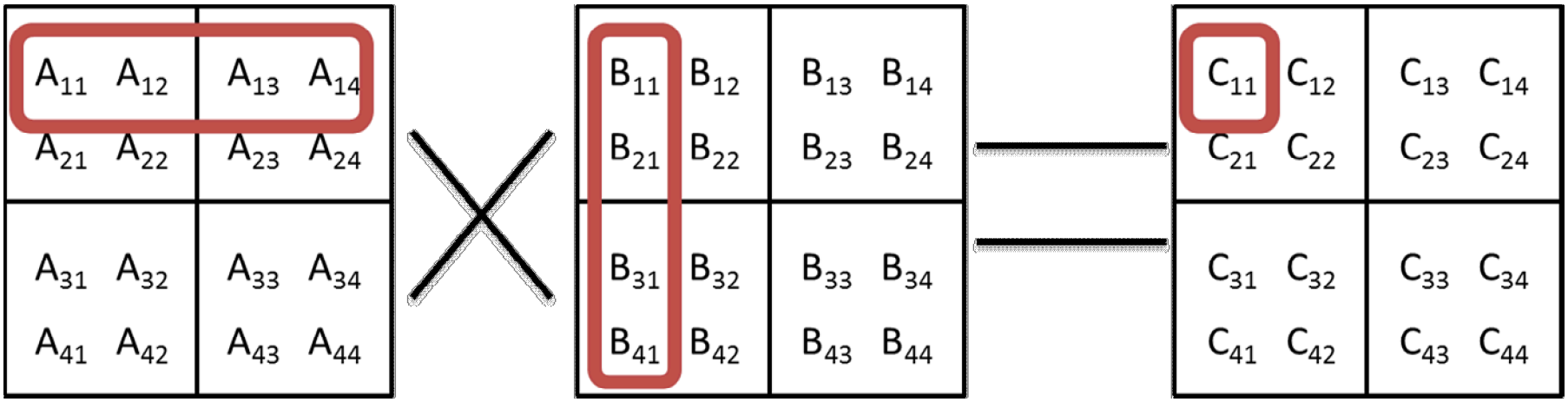
Backpropagation:

$$\Delta_k = \sum_k w_{jk}\delta_k$$

Graph Analysis:



$$\begin{pmatrix} 2 & 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \end{pmatrix}$$



Naïve algorithm for matrix multiplication is $O(N^3)$.

Strassen matrix multiplication

$$\left[\begin{array}{c|c} A_{11} & A_{12} \\ \hline A_{21} & A_{22} \end{array}\right] \times \left[\begin{array}{c|c} B_{11} & B_{12} \\ \hline B_{21} & B_{22} \end{array}\right] = \left[\begin{array}{c|c} C_{11} & C_{12} \\ \hline C_{21} & C_{22} \end{array}\right]$$

Standard

$$C_{11} = A_{11}B_{11} + A_{12}B_{21}$$
$$C_{12} = A_{11}B_{12} + A_{12}B_{22}$$
$$C_{21} = A_{21}B_{11} + A_{22}B_{21}$$
$$C_{22} = A_{21}B_{12} + A_{22}B_{22}$$

Strassen

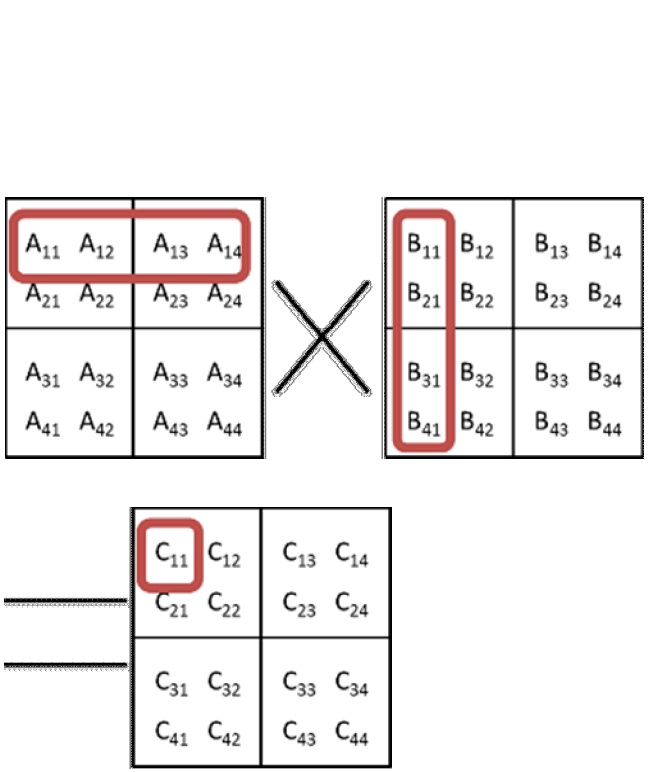
$$M_1 = (A_{11} + A_{22})(B_{11} + B_{22})$$
$$M_2 = (A_{21} + A_{22})B_{11}$$
$$M_3 = A_{11}(B_{12} - B_{22})$$
$$M_4 = A_{22}(B_{21} - B_{11})$$
$$M_5 = (A_{11} + A_{12})B_{22}$$
$$M_6 = (A_{21} - A_{11})(B_{11} + B_{12})$$
$$M_7 = (A_{12} - A_{22})(B_{21} + B_{22})$$

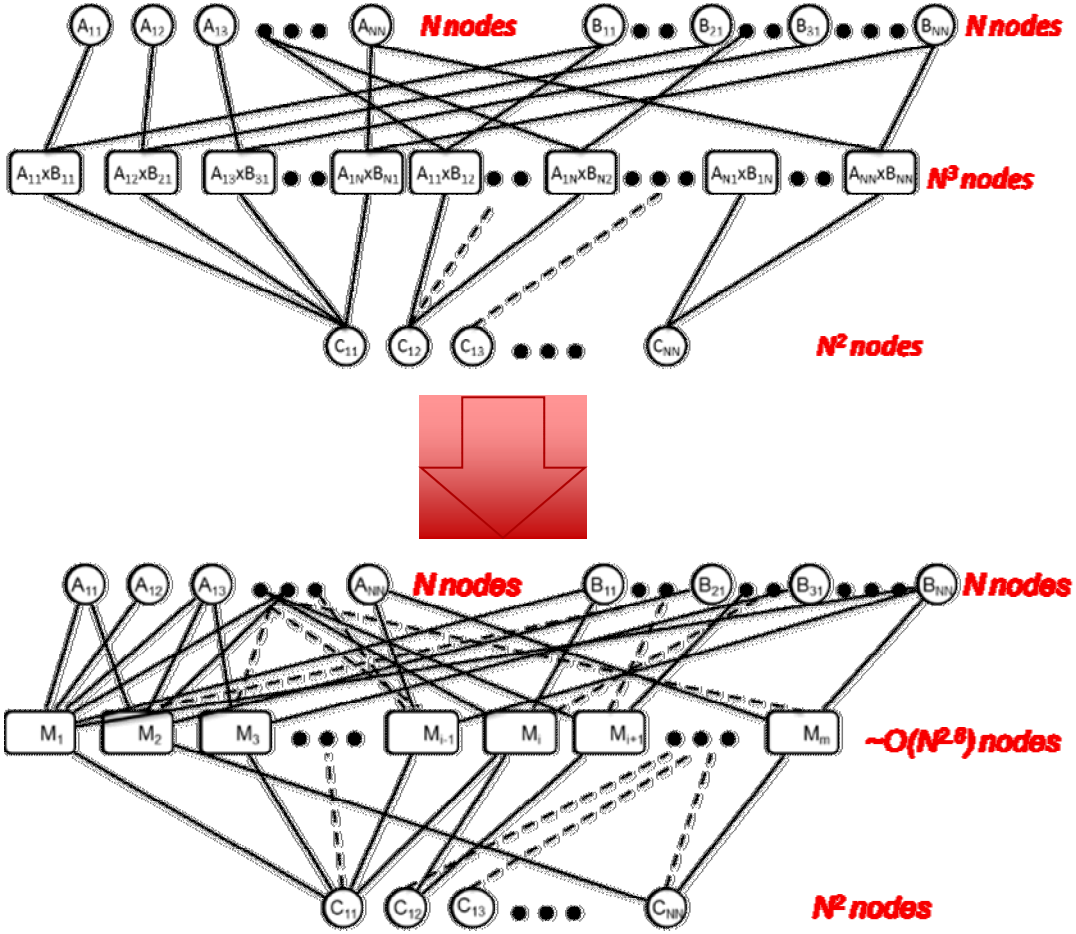
$$C_{11} = M_1 + M_4 - M_5 + M_7$$
$$C_{12} = M_3 + M_5$$
$$C_{21} = M_2 + M_4$$
$$C_{22} = M_1 - M_2 + M_3 + M_6$$

Standard: 8Ms, 4As → $O(N^3)$
Strassen: 7Ms, 18A/Ss→ $O(N^{2+\varepsilon})$

Strassen, Num Math, 1969

“Neural” network for matrix multiplication



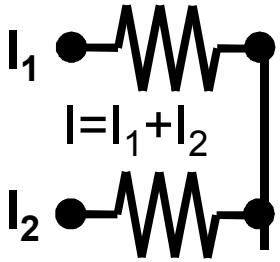


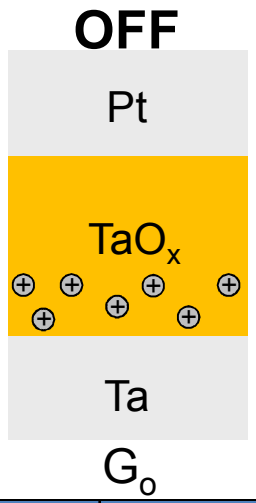
Strassen formulation of matrix multiply enables less than O(N^3) neurons – resulting in less power consumption

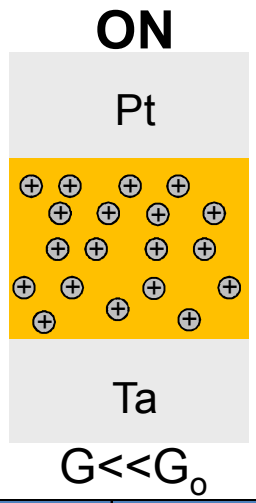
Resistive switching devices

$V = I \times R$

$I = G \times V$



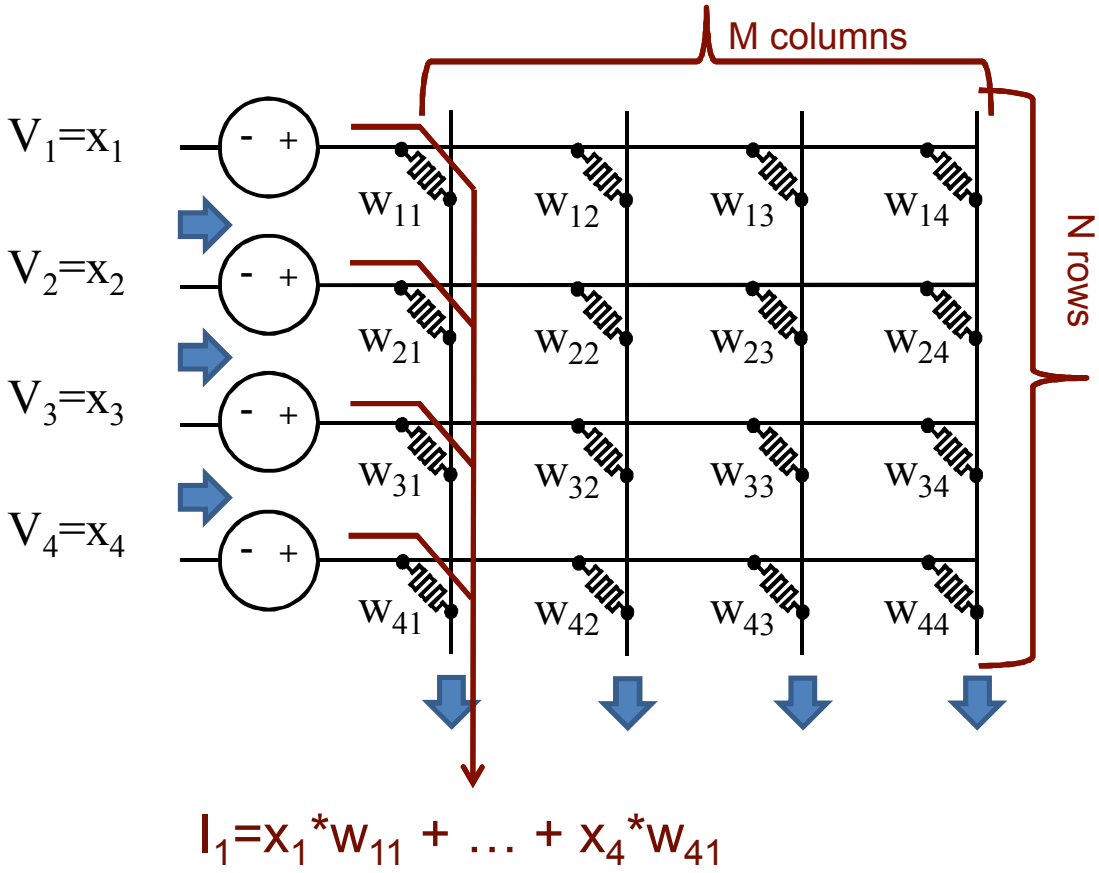




	DRAM	NAND Flash	PC-RAM	STT-MRAM	FeRAM	ReRAM	CBRAM
Maturity	Production (20 nm)	Production (16 nm)	Production (45 nm)	Production (65 nm)	Production (180 nm)	Production (180 nm)	Production (180nm)
Min device feature F (nm)	20	16	<10	16	28 nm	5	20 (5 est.)
Density (F^2)	6	10 (single layer)	4	8-20	22	4	4
Write Time (ns)	< 10	10000	50	13	<100	2	2
Write Energy (pJ/bit)	0.005	100	6	4	270	<1	<1
Endurance (W/E Cycles)	$>10^{16}$	10^4	$>10^9$	10^{12}	10^{14}	10^{12}	10^{10}
Retention	64 ms	1 - 10 y	> 10 y	weeks	> 10 y	> 10 y	> 10 y
Stackable	No	Yes	Yes	No	No	Yes	Yes
Process complexity	High/FE	High/FE	Low/BE	High/BE	High/BE	Low/BE	Low/BE

***many of these numbers are not universally agreed on

ReRAM is *O(N)* better than SRAM in energy consumption for vector-matrix multiply computations



SRAMs must fetch each vector per dot product ~ ***O(N²×M)***

Analog computation: multiplier and adder at each intersection; E ~ CV^2 ~ ***O(N×M)***

Agarwal et al., Front. Neuroscience 2016, 9, 484